

Appendices

- Appendix A** **Installing the Toolboxes**
- Appendix B** **Linear Algebra Refresher**
- Appendix C** **Geometry**
- Appendix D** **Lie Groups and Algebras**
- Appendix E** **Linearization, Jacobians and Hessians**
- Appendix F** **Solving Systems of Equations**
- Appendix G** **Gaussian Random Variables**
- Appendix H** **Kalman Filter**
- Appendix I** **Graphs**
- Appendix J** **Peak Finding**

A

Installing the Toolboxes

The Toolboxes are freely available from the book's home page

<http://www.petercorke.com/RVC>

which also has a lot of additional information related to the book such as web links (all those printed in the book and more), code, figures, exercises and errata.

Downloading and Installing

Two toolboxes support this book: the Robotics Toolbox (RTB) and the Machine Vision Toolbox (MVTB). For the second edition of this book the relevant versions are RTB v10 and MVTB v4.

Toolboxes can be installed from .zip or .mltbx format files, with details below. Once the toolboxes are downloaded you can explore their capability using

```
>> rtbdemo
```

or

```
>> mvtbdemo
```

From .mltbx File

Since MATLAB® R2014b toolboxes can be packaged as, and installed from, files with the extension .mltbx. Download the most recent version of [robot.mltbx](#) or [vision.mltbx](#) to your computer. Using MATLAB navigate to the folder where you downloaded the file and double-click it (or right-click then select Install). The Toolbox will be installed within the local MATLAB file structure, and the paths will be appropriately configured for this, and future MATLAB sessions.

From .zip File

Download the most recent version of [robot.zip](#) or [vision.zip](#) to your computer. Use your favorite unarchiving tool to unzip the files that you downloaded.

To add the Toolboxes to your MATLAB path execute the command

```
>> addpath RVCDIR ;  
>> startup_rvc
```

where RVCDIR is the full pathname of the directory where the folder `rvctools` was created when you unzipped the Toolbox files. The script `startup_rvc` adds various subfolders to your path and displays the version of the Toolboxes.

You will need to run the `startup_rvc` script each time you start MATLAB. Alternatively you can run `pathtool` and save the path configuration created by `startup_rvc`.

For installation from zip files, the files for both Toolboxes reside in a top-level directory called `rvctools` and beneath this are a number of subdirectories:

robot	The Robotics Toolbox.
vision	The Machine Vision Toolbox.
common	Utility functions common to the Robotics and Machine Vision Toolboxes.
simulink	Simulink® blocks for robotics and vision, as well as examples.
contrib	Code written by third-parties.

MEX-Files

Some functions in the Toolbox are implemented as MEX-files, that is, they are written in C for computational efficiency but are callable from MATLAB just like any other function. Source code is provided in the `mex` folder along with instructions and scripts to build the MEX-files from inside MATLAB or from the command line. You will require a C-compiler in order to build these files, but prebuilt MEX-files for a limited number of architectures are included.

Contributed Code

A number of useful functions are provided by third-parties and wrappers have been written to make them consistent with other Toolbox functions. If you attempt to access a contributed function that is not installed you will receive an error message.

The contributed code `contrib.zip` can be downloaded, expanded and then added your MATLAB path. If you installed the Toolboxes from `.zip` files then expand `contrib.zip` inside the folder `RVCDIR`.

Many of these contributed functions are part of active software projects and the downloadable file is a snapshot that has been tested and works as described in this book.

Getting Help

A Google group at <http://tiny.cc/rvcforum> provides answers to frequently asked questions, and has a user forum for discussing questions, issues and bugs.

License

All the non-third-party code is released under the LGPL license. This means you are free to distribute it in original or modified form provided that you keep the license and authorship information intact.

The third-party code modules are provided under various open-source licenses. The Toolbox compatibility wrappers for these modules are provided under compatible licenses.

MATLAB Versions

The Toolbox software for this book has been developed and tested using MATLAB R2015b and R2016a under Mac OS X (10.11 El Capitan). MATLAB continuously evolves so older versions of MATLAB are increasingly unlikely to work. Please do not report bugs if you are using a MATLAB version older than R2014a.

Octave

GNU Octave (www.octave.org) is an impressive piece of free software that implements a language that is close to, but not the same as, MATLAB. The Toolboxes will not work well with Octave, though with Octave 4 the incompatibilities are greatly reduced. An old version of the arm-robot functions described in Chap. 7–9 have been ported to Octave and this code is distributed in [RVCDIR/robot/octave](#).

B

Linear Algebra Refresher

B.1 Vectors

A rank 1 tensor.

We will only consider real vectors \blacktriangleleft which are an ordered n -tuple of real numbers v_1, v_2, \dots, v_n which is usually written as

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \text{ or } \mathbf{v} = (v_1, v_2, \dots, v_n)$$

which are a column- and row-vector respectively. These are equivalent to an $n \times 1$ and a $1 \times n$ matrix respectively, and can be multiplied with a conforming matrix.

The numbers v_1, v_2 etc. are called the scalar components of \mathbf{v} , and v_i is called the i^{th} component of \mathbf{v} . For a 3-vector we often write the elements as $\mathbf{v} = (v_x, v_y, v_z)$.

The symbol \mathbb{R}^n represents the set of ordered n -tuples of real numbers, each vector is a point in this space, that is $\mathbf{v} \in \mathbb{R}^n$. The elements of \mathbb{R}^2 can be represented in a plane by a point or a directed line segment. The elements of \mathbb{R}^3 can be represented in a volume by a point or a directed line segment.

A vector space is an n -dimensional space whose elements are vectors *plus* the operations of addition and scalar multiplication. The addition of any two elements $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ yields $(a_1 + b_1, a_2 + b_2, \dots, a_n + b_n)$ and $s\mathbf{a} = (sa_1, sa_2, \dots, sa_n)$. Both results are element of \mathbb{R}^n . The negative of a vector is obtained by negating each element of the vector $-\mathbf{a} = (-a_1, -a_2, \dots, -a_n)$.

We can use a vector to represent a point with coordinates (x_1, x_2, \dots, x_n) which is called a coordinate vector. However we need to be careful because the operations of addition and scalar multiplication, while valid for vectors are meaningless for points. We can add a vector to the coordinate vector of a point to obtain the coordinate vector of another point, and we can subtract one coordinate vector from another, and the result is the displacement between the points.

The magnitude or length of a vector is a nonnegative scalar given by its p -norm

$$\|\mathbf{v}\|_p = \left(\sum_{i=1}^n |v_i|^p \right)^{1/p}$$

The Euclidean length of a vector is given by $\|\mathbf{v}\|_2$ which is also referred to as the L_2 norm and is generally assumed when p is omitted, for example $\|\mathbf{v}\|$. A unit vector is one where $\|\mathbf{v}\|_2 = 1$ and is denoted as $\hat{\mathbf{v}}$. The L_1 norm is sum of the absolute value of the elements and is also known as the Manhattan distance, it is the distance traveled when confined to moving along the lines in a grid. The L_∞ norm is the maximum element of the vector.

The dot product of two column vectors is a scalar

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a} = \mathbf{a}^T \mathbf{b} = \mathbf{b}^T \mathbf{a} = \sum_{i=1}^n a_i b_i = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2 \cos \theta$$

where θ is the angle between the vectors. $\mathbf{a} \cdot \mathbf{b} = 0$ when the vectors are orthogonal. For 3-vectors the cross product

$$\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a} = \det \begin{pmatrix} \hat{x} & \hat{y} & \hat{z} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{pmatrix} = [\mathbf{a}]_{\times} \mathbf{b} = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2 \sin \theta \hat{n}$$

where \hat{x} is a unit-vector parallel to the x-axis etc., $[\cdot]_{\times}$ is a skew-symmetric matrix as described in the next section, and \hat{n} is a unit-vector normal to the plane containing \mathbf{a} and \mathbf{b} . If the vectors are parallel $\mathbf{a} \times \mathbf{b} = 0$.

B.2 Matrices

A taxonomy of matrices is shown in Fig. B.1. In this book we are concerned only with real $m \times n$ matrices ▶

$$\mathbf{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}, \mathbf{A} \in \mathbb{R}^{m \times n}$$

with m rows and n columns. If $n = m$ the matrix is square.

The transpose is

$$\mathbf{B} = \mathbf{A}^T, b_{i,j} = a_{j,i} \quad \forall i, j$$

and it can be shown that

$$(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T, (\mathbf{ABC})^T = \mathbf{C}^T \mathbf{B}^T \mathbf{A}^T, \text{ etc.}$$

Real matrices are a subset of all matrices. For the general case of complex matrices the term Hermitian is the analog of symmetric, and unitary the analog of orthogonal. \mathbf{A}^H denotes the Hermitian transpose, the complex conjugate transpose of the complex matrix \mathbf{A} . Matrices are rank 2 tensors.

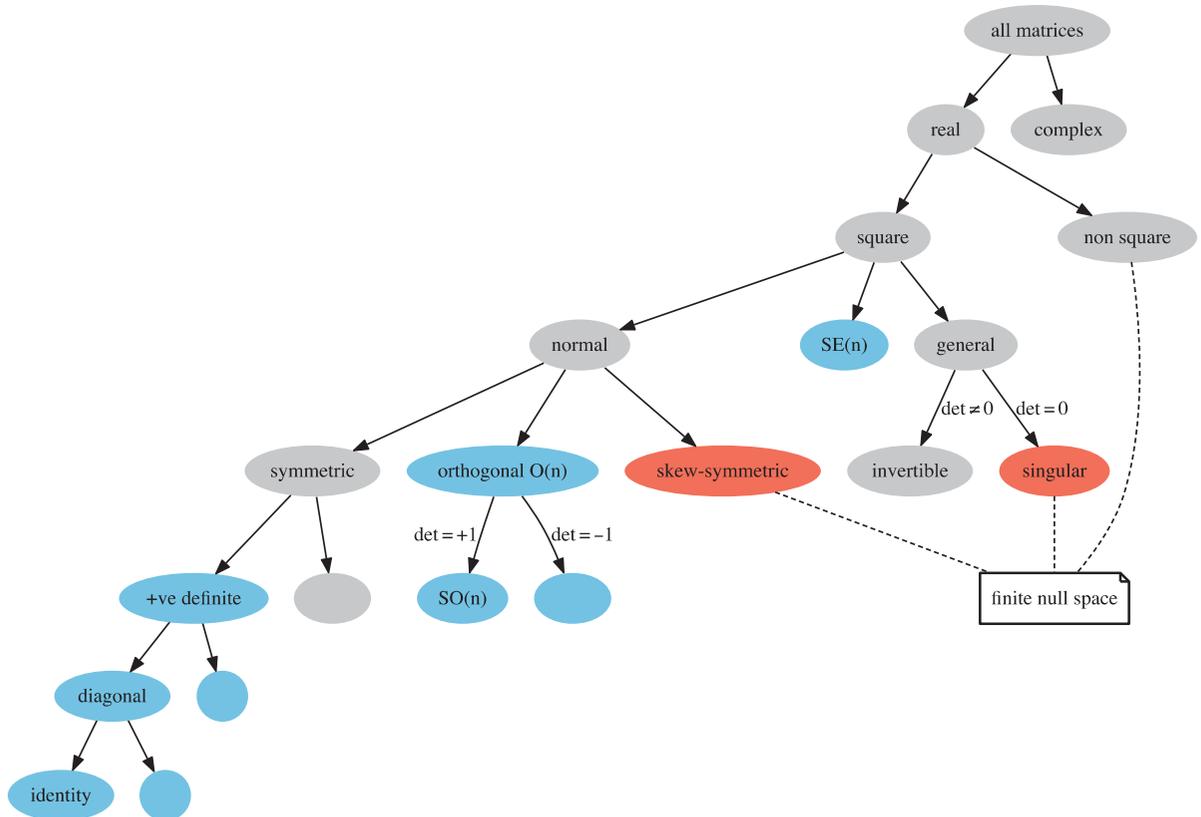


Fig. B.1. Taxonomy of matrices. Classes of matrices that are always singular are shown in red, those that are never singular are shown in blue

B.2.1 Square Matrices

A square matrix may have an inverse A^{-1} in which case

$\mathbf{Ai} = \mathbf{inv}(\mathbf{A}) \quad \mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_{n \times n}$

where

$$\mathbf{I}_{n \times n} = \begin{pmatrix} 1 & & & \mathbf{0} \\ & 1 & & \\ & & \ddots & \\ \mathbf{0} & & & 1 \end{pmatrix} \in \mathbb{R}^{n \times n}$$

is the identity matrix, a unit diagonal matrix. The inverse exists provided that the matrix is nonsingular, that is, its determinant $\det(A) \neq 0$. The inverse can be computed from the matrix of cofactors. If A and B are square and nonsingular then

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}, \quad (\mathbf{ABC})^{-1} = \mathbf{C}^{-1}\mathbf{B}^{-1}\mathbf{A}^{-1}, \quad \text{etc.}$$

and also

$$(\mathbf{A}^T)^{-1} = (\mathbf{A}^{-1})^T$$

The inverse can be written as

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \text{adj}(\mathbf{A})$$

where $\text{adj}(A)$ is the transpose of the matrix of cofactors and known as the adjugate or adjoint matrix and sometimes denoted by A^* . If $B = \text{adj}(A)$ then $A = \text{adj}(B)$. If A is nonsingular the adjugate can be computed by

$$\text{adj}(A) = \det(A)A^{-1}$$

For a square matrix if

$\mathbf{A} = \mathbf{A}^T$ the matrix is **symmetric**. The inverse of a symmetric matrix is also symmetric. Many matrices that we encounter in robotics are symmetric, for example covariance matrices and manipulator inertia matrices.

$\mathbf{S} = \mathbf{skew}(\mathbf{v}) \quad \mathbf{A} = -\mathbf{A}^T$ the matrix is **skew-symmetric** or **anti-symmetric**. Such a matrix has a zero diagonal, is always singular and has the property that $[\mathbf{a}\mathbf{v}]_x = a[\mathbf{v}]_x$, $[\mathbf{R}\mathbf{v}]_x = \mathbf{R}[\mathbf{v}]_x$ and $\mathbf{v}^T[\mathbf{v}]_x = [\mathbf{v}]_x \mathbf{v} = 0, \forall \mathbf{v}$. For the 3×3 case

$$\mathbf{S} = [\mathbf{v}]_x = \begin{pmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{pmatrix} \tag{B.1}$$

$\mathbf{v} = \mathbf{vex}(\mathbf{S})$

and the inverse operation is

$$\mathbf{v} = \mathbf{vex}(\mathbf{S})$$

$\mathbf{A}^{-1} = \mathbf{A}^T$ the matrix is **orthogonal**. The matrix is also known as orthonormal since its column vectors (and row vectors) must be of unit length and orthogonal to each other. The product of two orthogonal

matrices of the same size is also an orthogonal matrix. The set of $n \times n$ orthogonal matrices forms a group $O(n)$, known as the orthogonal group. The determinant of an orthogonal matrix is either $+1$ or -1 . The subgroup $SO(n)$ consisting of orthogonal matrices with determinant $+1$ is called the special orthogonal group. The columns (and rows) are orthogonal vectors, that is, their dot product is zero.

$A^T A = A A^T$ the matrix is **normal** and can be diagonalized by an orthogonal matrix U so that $U^T A U$ is a diagonal matrix. All symmetric, skew-symmetric and orthogonal matrices are normal matrices as are matrices of the form $A = B^T B = B B^T$ where B is an arbitrary matrix.

The square matrix $A \in \mathbb{R}^{n \times n}$ can be applied as a linear transformation to a vector $x \in \mathbb{R}^n$

$$x' = Ax$$

which results in another vector, generally with a change in its length and direction. However there are some important special cases. If $A \in SO(n)$ the transformation is isometric and the vector's *length* is unchanged $\|x'\| = \|x\|$.

In 2-dimensions if x is the set of all points lying on a circle then x' defines points that lie on an ellipse. The MATLAB® builtin demonstration

```
>> eigshow
```

shows this very clearly as you interactively drag the tip of the vector x around the unit circle.

The eigenvectors of a square matrix are those vectors x such that

```
[x,e] = eig(A)
```

$$Ax = \lambda_i x \tag{B.2}$$

that is, their direction is unchanged when transformed by the matrix. They are simply scaled by λ_i , the corresponding eigenvalue. The matrix has n eigenvalues (the *spectrum* of the matrix) which can be real or complex. For an orthogonal matrix the eigenvalues lie on a unit circle in the complex plane, $|\lambda_i| = 1$, and the eigenvectors are all orthogonal to one another.

The eigenvalues of a real symmetric matrix are all real and we classify the matrix according to the sign of its eigenvalues

- $\lambda_i > 0, \forall i$ positive definite
- $\lambda_i \geq 0, \forall i$ positive semi-definite
- $\lambda_i < 0, \forall i$ negative definite
- otherwise indefinite

The inverse of a positive definite matrix is also positive definite.

The matrices $A^T A$ and $A A^T$ are always symmetric and positive semidefinite. This implies than any symmetric matrix A can be written as

$$A = LL^T$$

where L is the Cholesky decomposition of A .

```
L = chol(A)
```

The matrix R such that

$$A = RR$$

```
R = sqrtm(A)
```

is the square root of A or $A^{1/2}$.

If T is any nonsingular matrix then

$$A = TBT^{-1}$$

is known as a similarity transform and A and B are said to be similar, and it can be shown that the eigenvalues are unchanged by the transformation.

If A is nonsingular then the eigenvectors of A^{-1} are the same as A and the eigenvalues of A^{-1} are the reciprocal of those of A . The eigenvalues of A^T are the same as those of A but the eigenvectors are different.

The matrix form of Eq. B.2 is

$$AX = X\Lambda$$

where $X \in \mathbb{R}^{n \times n}$ is a matrix of eigenvectors of A , arranged column-wise, and Λ is a diagonal matrix of corresponding eigenvalues. If X is not singular we can rearrange this as

$$A = X\Lambda X^{-1}$$

which is the eigenvalue or spectral decomposition of the matrix. This implies that the matrix can be diagonalized by a similarity transform

$$\Lambda = X^{-1}AX$$

If A is symmetric then X is orthogonal and we can instead write

$$A = X\Lambda X^T \tag{B.3}$$

det (A) The determinant of a square matrix $A \in \mathbb{R}^{n \times n}$ is the factor by which the transformation changes volumes in an n -dimensional space. For 2-dimensions imagine a shape defined by points x_i with an enclosed area a . The shape formed by the points Ax_i would have an enclosed area $a \det(A)$. If A is singular the points Ax_i would lie at a single point or along a line and have zero enclosed area. In a similar way for 3-dimensions, the determinant is a scale factor applied to the volume of a set of points mapped through the transformation A .

The determinant is equal to the product of the eigenvalues

$$\det(A) = \prod_{i=1}^n \lambda_i$$

trace (A) thus a matrix with one or more zero eigenvalues will be singular. A positive definite matrix, $\lambda_i > 0$, therefore has $\det(A) > 0$ and is not singular. The trace of a matrix is the sum of the diagonal elements

$$\text{tr}(A) = \sum_{i=1}^n A_{ii}$$

which is also the sum of the eigenvalues

$$\text{tr}(A) = \sum_{i=1}^n \lambda_i$$

The columns of $A = (c_1 c_2 \cdots c_n)$ can be considered as a set of vectors that define a space – the column space. Similarly, the rows of A can be considered as a set of vectors that define a space – the row space. The column rank of a matrix is the number of linearly independent columns of A . Similarly, the row rank is the number of linearly

where $r = \text{rank}(A)$ is the rank of A and $\sigma_i \geq \sigma_{i+1}$. For the case where $r < n$ the diagonal will have zero elements as shown. Columns of V^T corresponding to the zero columns of Σ define the null space of A . The condition number of a matrix A is $\max \sigma_i / \min \sigma_i$ and a high value means the matrix is close to singular or “poorly conditioned”.

cond(A)

The matrix quadratic form

$$s = \mathbf{x}^T A \mathbf{x} \tag{B.4}$$

is a scalar. If A is positive definite then $s = \mathbf{x}^T A \mathbf{x} > 0, \forall \mathbf{x} \neq 0$.

For the case that A is diagonal this can be written

$$s = \sum_{i=1}^n A_{ii} x_i^2$$

which is a weighted sum of squares. If A is symmetric then

$$s = \sum_{i=1}^n A_{ii} x_i^2 + 2 \sum_{i=1}^n \sum_{j=i+1}^n A_{ij} x_i x_j$$

the result also includes products or correlations between elements of \mathbf{x} .

The Mahalanobis distance is a weighted distance or norm

$$s = \sqrt{\mathbf{v}^T \mathbf{P}^{-1} \mathbf{v}}$$

where $\mathbf{P} \in \mathbb{R}^{n \times n}$ is a covariance matrix which down-weights components of \mathbf{v} where uncertainty is high.

C

Geometry

Geometric concepts such as points, lines, ellipses and planes are critical to the fields of robotics and robotic vision. We briefly summarize key representations in both Euclidean and projective (homogeneous coordinate) space.

C.1 Euclidean Geometry

C.1.1 Points

A point in n -dimensional space is represented by an n -tuple, an ordered set of n numbers $(x_1, x_2 \cdots x_n)$ which define the coordinates of the point. The tuple can also be interpreted as a vector – a coordinate vector – from the origin to the point.

C.1.2 Lines

C.1.2.1 Lines in 2D

A line is defined by $\ell = (a, b, c)$ such that

$$ax + by + c = 0 \quad (\text{C.1})$$

which is a generalization of the line equation we learned in school $y = mx + c$ but which can easily represent a vertical line by setting $a = 0$. $v = (a, b)$ is a vector parallel to the line, and $v = (-b, a)$ is a vector normal to the line. The line that joins two points is given by the solution to

$$\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = 0$$

which is found from the right-null space of the left-most term. The intersection point of two lines is

$$\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = - \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

which has no solution if the lines are parallel – the left-most term is singular.

We can also represent the line in polar form

$$\cos\theta x + \sin\theta y + \rho = 0$$

where θ is the angle from the x -axis to the line and ρ is the normal distance between the line and the origin, as shown in Fig. 13.18.

C.1.2.2 Lines in 3D and Plücker Coordinates

We can define a line by two points, \mathbf{p} and \mathbf{q} , as shown in Fig. C.1, which would require a total of six parameters $\ell = (q_x, q_y, q_z, p_x, p_y, p_z)$. However since these points can be arbitrarily chosen there would be an infinite set of parameters that represent the same line making it hard to determine the equivalence of two lines.

There are advantages in representing a line as

$$\ell = (\boldsymbol{\omega} \times \mathbf{q}, \mathbf{p} - \mathbf{q}) = (\mathbf{v}, \boldsymbol{\omega}) \in \mathbb{R}^6$$

where $\boldsymbol{\omega}$ is the direction of the line and \mathbf{v} is the moment of the line – a vector from the origin to a point on the line and normal to the line. This is a Plücker coordinate vector – a six dimensional quantity subject to two constraints: the coordinates are homogeneous and thus invariant to overall scale factor; and $\mathbf{v} \cdot \boldsymbol{\omega} = 0$. Lines therefore have 4 degrees-of-freedom and the Plücker coordinates lie on a 4-dimensional manifold in 6-dimensional space. Lines with $\boldsymbol{\omega} = 0$ lie at infinity and are known as ideal lines. ▶

In MATLAB® we will first define two points as column vectors

```
>> P = [2 3 4]'; Q = [3 5 7]';
```

and then create a Plücker line object

```
>> L = Plucker(P, Q)
L =
{ 1 -2 1; -1 -2 -3 }
```

which displays the \mathbf{v} and $\boldsymbol{\omega}$ components. These can be accessed as properties

```
>> L.v'
ans =
     1     -2     1
>> L.w'
ans =
    -1    -2    -3
```

A Plücker line can also be represented as a skew-symmetric matrix

```
>> L.L
ans =
     0     1     2    -1
    -1     0     1    -2
    -2    -1     0    -3
     1     2     3     0
```

which can also be formed by $\tilde{\mathbf{p}}\tilde{\mathbf{q}}^T - \tilde{\mathbf{q}}\tilde{\mathbf{p}}^T$.

To plot this line we first define a region of 3D space ▶ then plot it in blue

```
>> axis([-5 5 -5 5 -5 5]);
>> L.plot('b');
```

The line is the set of all points

$$\mathbf{p}(\lambda) = \frac{\mathbf{v} \times \boldsymbol{\omega}}{\boldsymbol{\omega} \cdot \boldsymbol{\omega}} + \lambda \boldsymbol{\omega}, \quad \lambda \in \mathbb{R}$$

which can be generated parametrically in terms of a scalar parameter

```
>> L.point([0 1 2])
ans =
    -0.5714    -1.5714    -2.5714
    -0.1429    -2.1429    -4.1429
     0.2857    -2.7143    -5.7143
```

where the columns are points on the line corresponding to $\lambda = 0, 1, 2$.

This is not intuitive but consider two parallel planes and an arbitrary 3D line passing through them. The line can be described by the 2-dimensional coordinates of its intersection point on each plane – a total of four coordinates.

Ideal as in imaginary, not as in perfect.

Since lines are infinite we need to specify a finite volume in which to draw it.

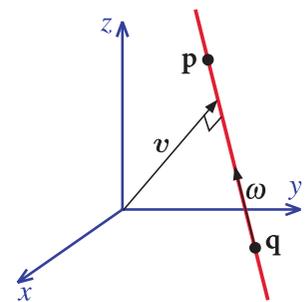


Fig. C.1. Describing a line in 3-dimensions

A point x is closest to the line when

$$\lambda = \frac{(x - q) \cdot \omega}{\omega \cdot \omega}$$

For the point (1, 2, 3) the closest point on the line, and its distance, is given by

```
>> [x, d] = L.closest([1 2 3]')
x =
    3.1381
    2.5345
    1.9310
d =
    2.4495
```

The line intersects the plane $n^T x + d = 0$ at the point coordinate

$$x = \frac{v \times n - d\omega}{\omega \cdot n}$$

For the xy -plane the line intersects at

```
>> L.plane_intersect([0 0 1 0])'
ans =
    0.6667    0.3333    0
```

Two lines can be identical, coplanar or skewed. Identical lines have linearly dependent Plücker coordinates, that is, $\ell_1 = \lambda \ell_2$. If coplanar they can be parallel or intersecting and if skewed can be intersecting or not. If lines have $\omega_1 \times \omega_2 = 0$ they are parallel otherwise they are skewed.

The minimum distance between two lines is

$$d = \omega_1 \cdot v_2 + \omega_2 \cdot v_1$$

and is zero if they intersect.

The side operator is a permuted dot product

$$\text{side}(\ell^1, \ell^2) = \ell_1^1 \ell_5^2 + \ell_2^1 \ell_6^2 + \ell_3^1 \ell_4^2 + \ell_4^1 \ell_3^2 + \ell_5^1 \ell_1^2 + \ell_6^1 \ell_2^2$$

which is zero if the lines intersect or are parallel and is computed by the `side` method.

We can transform a Plücker line by the adjoint of a rigid-body motion.

$$\ell' = \text{Ad}(\xi)\ell$$



Julius Plücker (1801–1868) was a German mathematician and physicist who made contributions to the study of cathode rays and analytical geometry. He was born at Elberfeld and studied at Düsseldorf, Bonn, Heidelberg and Berlin and went to Paris in 1823 where he was influenced by the French geometry movement. In 1825 he returned to the University of Bonn, was made professor of mathematics in 1828, and professor of physics in 1836. In 1858 he proposed that the lines of the spectrum, discovered by his colleague Heinrich Geissler (of Geissler tube fame), were characteristic of the chemical substance which emitted them. In 1865, he returned to geometry and invented what was known as line geometry. He was the recipient of the Copley Medal from the Royal Society in 1866, and is buried in the Alter Friedhof (Old Cemetery) in Bonn.

C.1.3 Planes

A plane is defined by a 4-vector $\pi = (a, b, c, d)$ such that

$$ax + by + cz + d = 0$$

which can be written in point-normal form as

$$\mathbf{n}^T(\mathbf{x} - \mathbf{p}) = 0$$

for a plane containing a point with coordinate \mathbf{p} and a normal \mathbf{n} , or more generally as

$$\mathbf{n}^T \mathbf{x} + d = 0$$

A plane can be defined by 3 points

$$\begin{pmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \end{pmatrix} \pi = 0$$

and solved for using the right-null space of the left-most term, or by two nonparallel lines

$$\pi = (\boldsymbol{\omega}_1 \times \boldsymbol{\omega}_2, \mathbf{v}_1 \cdot \boldsymbol{\omega}_2)$$

or by a line and a point with coordinate \mathbf{r}

$$\pi = (\boldsymbol{\omega} \times \mathbf{r} - \mathbf{v}, \mathbf{v} \cdot \mathbf{r})$$

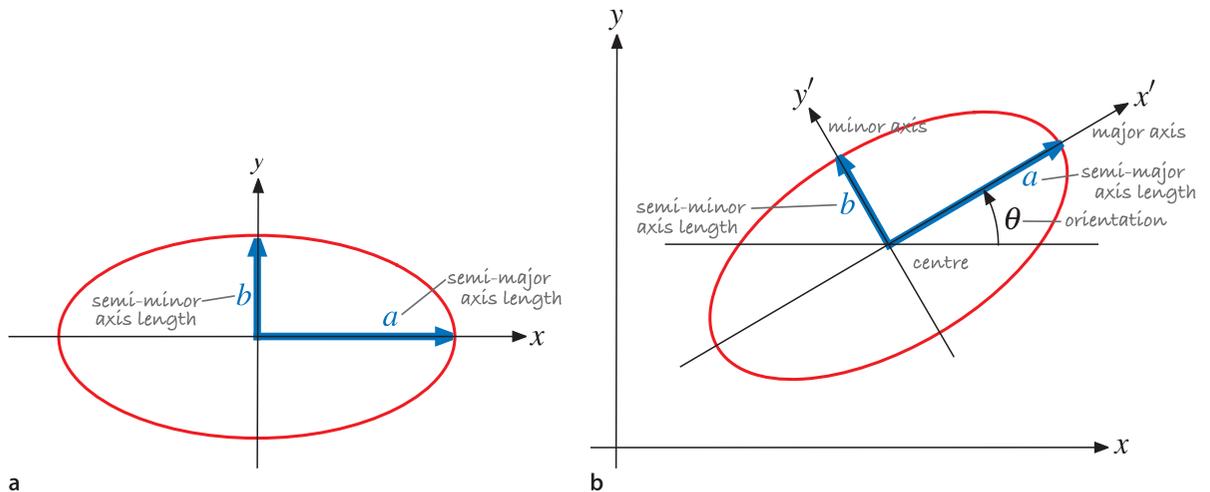
A point is defined as the intersection point of three planes

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = - \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix}$$

The Plücker line formed by the intersection of two planes is

$$\ell = (\mathbf{n}_1 \times \mathbf{n}_2, d_2 \mathbf{n}_1 - d_1 \mathbf{n}_2)$$

Fig. C.2. Ellipses. **a** Canonical ellipse centered at the origin and aligned with the x - and y -axes; **b** general form of ellipse



C.1.4 Ellipses and Ellipsoids

An ellipse belongs to the family of planar curves known as conics. The simplest form of an ellipse is defined implicitly

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

and is shown in Fig. C.2a. This canonical ellipse is centered at the origin and has its major and minor axes aligned with the x - and y -axes. The radius in the x -direction is a and in the y -direction is b . The longer of the two radii is known as the semi-major axis length and the other is the semi-minor axis length.

We can write the ellipse in matrix quadratic form Eq. B.4 as

$$\begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} 1/a^2 & 0 \\ 0 & 1/b^2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = 1$$

$$\mathbf{x}^T \begin{pmatrix} a^2 & 0 \\ 0 & b^2 \end{pmatrix}^{-1} \mathbf{x} = 1 \quad (\text{C.2})$$

$$\mathbf{x}^T \mathbf{E}^{-1} \mathbf{x} = 1 \quad (\text{C.3})$$

In the most general form \mathbf{E} is a symmetric matrix

$$\mathbf{E} = \begin{pmatrix} A & C \\ C & B \end{pmatrix} \quad (\text{C.4})$$

and its determinant $\det(\mathbf{E}) = AB - C^2$ defines the type of conic

$$\det(\mathbf{E}) \begin{cases} > 0 & \text{ellipse} \\ = 0 & \text{parabola} \\ < 0 & \text{hyperbola} \end{cases}$$

An ellipse is therefore represented by a positive definite symmetric matrix \mathbf{E} . Conversely any positive definite symmetric matrix, such as an inertia matrix or covariance matrix, can be represented by an ellipse.

Nonzero values of C change the orientation of the ellipse. The ellipse can be arbitrarily centered at \mathbf{x}_c by writing it in the form

$$(\mathbf{x} - \mathbf{x}_c)^T \mathbf{E}^{-1} (\mathbf{x} - \mathbf{x}_c) = 1$$

which leads to the general ellipse shown in Fig. C.2b.

Since \mathbf{E} is symmetric it can be diagonalized by Eq. B.3

$$\mathbf{E} = \mathbf{X} \mathbf{\Lambda} \mathbf{X}^T$$

where \mathbf{X} is an orthogonal matrix comprising the eigenvectors of \mathbf{E} . The inverse is

$$\mathbf{E}^{-1} = \mathbf{X} \mathbf{\Lambda}^{-1} \mathbf{X}^T$$

so the quadratic form becomes

$$\begin{aligned} \mathbf{x}^T \mathbf{X} \mathbf{\Lambda}^{-1} \mathbf{X}^T \mathbf{x} &= 1 \\ (\mathbf{X}^T \mathbf{x})^T \mathbf{\Lambda}^{-1} (\mathbf{X}^T \mathbf{x}) &= 1 \\ \mathbf{x}'^T \mathbf{\Lambda}^{-1} \mathbf{x}' &= 1 \end{aligned}$$

This is similar to Eq. C.3 but with the ellipse defined by the diagonal matrix Λ with respect to the rotated coordinated frame $\mathbf{x}' = \mathbf{X}^T \mathbf{x}$. The major and minor ellipse axes are aligned with the eigenvectors of E . The squared radii of the ellipse are the eigenvalues of E or the diagonal elements of Λ .

For the general case of $E \in \mathbb{R}^{n \times n}$ the result is an ellipsoid in n -dimensional space. The Toolbox function `plot_ellipse` will draw an ellipse for the $n = 2$ case and an ellipsoid for the $n = 3$ case.

Alternatively the ellipse can be represented in polynomial form by writing as

$$(\mathbf{x} - (x_0, y_0))^T \begin{pmatrix} a & c \\ c & b \end{pmatrix} (\mathbf{x} - (x_0, y_0)) = 1$$

and expanding to obtain

$$e_1 x^2 + e_2 y^2 + e_3 xy + e_4 x + e_5 y + e_6 = 0$$

where $e_1 = a$, $e_2 = b$, $e_3 = 2c$, $e_4 = -2(ax_0 + cy_0)$, $e_5 = -2(by_0 + cx_0)$ and $e_6 = ax_0^2 + by_0^2 + 2cx_0y_0 - 1$. The ellipse has only five degrees of freedom, its center coordinate and the three unique elements in E . For a nondegenerate ellipse where $e_1 \neq 0$ we can rewrite the polynomial in normalized form

$$x^2 + E_1 y^2 + E_2 xy + E_3 x + E_4 y + E_5 = 0 \quad (\text{C.5})$$

with five unique parameters.

C.1.4.1 Properties

The area of an ellipse is πab and its eccentricity is

$$\varepsilon = \frac{\sqrt{a^2 - b^2}}{a}$$

The eigenvectors of E define the principal directions of the ellipse and the square root of the eigenvalues are the corresponding radii.

Consider the ellipse

$$\mathbf{x} \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix}^{-1} \mathbf{x} = 1$$

which is represented in MATLAB by

```
>> E = [2 -1; -1 1];
```

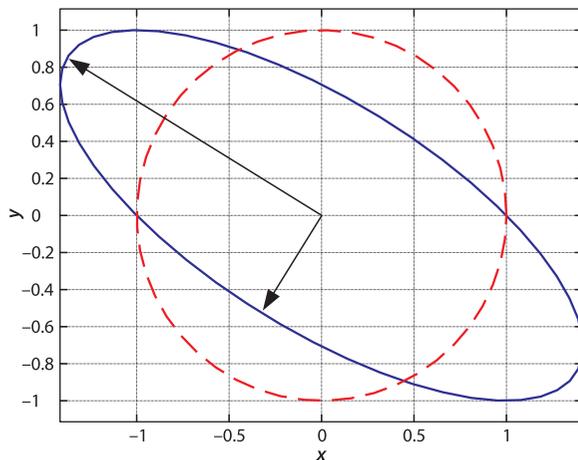


Fig. C.3. Ellipse corresponding to a symmetric 2×2 matrix, and the unit circle shown in red. The arrows indicate the major and minor axes of the ellipse

We can plot this by

```
>> plot_ellipse(E)
```

which is shown in Fig. C.3.

The eigenvectors and eigenvalues of E are

```
>> [x,e] = eig(E)
x =
   -0.5257   -0.8507
   -0.8507    0.5257
e =
    0.3820    0
         0    2.6180
```

and the ellipse radii are

```
>> r = sqrt(diag(e))
r =
    0.6180
    1.6180
```

which correspond to b and a respectively. If either radius is equal to zero the ellipse is degenerate and becomes a line. If both radii are zero the ellipse is a point.

The eigenvectors are unit vectors in the minor- and major-axis directions and we will scale them by the radii to yield radius vectors which we can plot

```
>> arrow([0 0]', x(:,1)*r(1));
>> arrow([0 0]', x(:,2)*r(2));
```

The orientation of the ellipse is the angle of the major-axis with respect to the horizontal axis and is

$$\theta = \tan^{-1} \frac{x_y}{x_x}$$

For our example this is

```
>> atan2(x(2,2), x(1,2)) * 180/pi
ans =
    148.2825
```

in units of degrees.

The ellipse area is $\pi r_1 r_2$ and the ellipsoid volume is $\frac{4}{3}\pi r_1 r_2 r_3$ where the radii $r_i = \sqrt{\lambda_i}$ where λ_i are the eigenvalues of E . Since $\det(E) = \Pi \lambda_i$ the area or volume is proportional to $\sqrt{\det(E)}$.

C.1.4.2 Drawing an Ellipse

In order to draw an ellipse we first define a point coordinate $\mathbf{y} = [x, y]^T$ on the unit circle

$$\mathbf{y}^T \mathbf{y} = 1$$

and rewrite Eq. C.3 as

$$\mathbf{x}^T \mathbf{E}^{-\frac{1}{2}} \mathbf{E}^{-\frac{1}{2}} \mathbf{x} = 1$$

where $\mathbf{E}^{\frac{1}{2}}$ is the matrix square root (MATLAB function `sqrtn`). Equating these two equations we can write

$$\mathbf{x}^T \mathbf{E}^{-\frac{1}{2}} \mathbf{E}^{-\frac{1}{2}} \mathbf{x} = \mathbf{y}^T \mathbf{y}$$

It is clear that

$$\mathbf{y} = \mathbf{E}^{-\frac{1}{2}}\mathbf{x}$$

which we can rearrange as

$$\mathbf{x} = \mathbf{E}^{\frac{1}{2}}\mathbf{y}$$

which transforms a point on the unit circle to a point on an ellipse. If the ellipse is centered at \mathbf{x}_c rather than the origin we can perform a change of coordinates

$$(\mathbf{x} - \mathbf{x}_c)^T \mathbf{E}^{-\frac{1}{2}} \mathbf{E}^{-\frac{1}{2}} (\mathbf{x} - \mathbf{x}_c) = 1$$

from which we write the transformation as

$$\mathbf{x} = \mathbf{E}^{\frac{1}{2}}\mathbf{y} + \mathbf{x}_c$$

Continuing the MATLAB example above

```
>> E = [2 -1; -1 1];
```

We define a set of points on the unit circle

```
>> th = linspace(0, 2*pi, 50);
>> y = [cos(th); sin(th)];
```

which we transform to points on the perimeter of the ellipse

```
>> x = (sqrtm(E) * y)';
>> plot(x(:,1), x(:,2));
```

which is encapsulated in the Toolbox function

```
>> plot_ellipse(E, [0 0])
```

An ellipsoid is described by a positive-definite symmetric 3×3 matrix. Drawing an ellipsoid is tackled in an analogous fashion and `plot_ellipse` is also able to display a 3-dimensional ellipsoid.

C.1.4.3 Fitting an Ellipse to Data

From a Set of Interior Points

We wish to find the equation of an ellipse that best fits a set of points that lie within the ellipse boundary. A common approach is to find the ellipse that has the same mass properties as the set of points. From the set of N points $\mathbf{x}_i = (x_i, y_i)$ we can compute the moments

$$\begin{aligned} m_{00} &= N \\ m_{10} &= \sum_{i=1}^N x_i \\ m_{01} &= \sum_{i=1}^N y_i \end{aligned}$$

The center of the ellipse is taken to be the centroid of the set of points

$$(x_c, y_c) = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right)$$

which allows us to compute the central second moments

$$\begin{aligned}\mu_{20} &= \sum_{i=1}^N (x_i - x_c)^2 \\ \mu_{02} &= \sum_{i=1}^N (y_i - y_c)^2 \\ \mu_{11} &= \sum_{i=1}^N (x_i - x_c)(y_i - y_c)\end{aligned}$$

The inertia matrix for a general ellipse is the symmetric matrix

$$J = \begin{pmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{pmatrix}$$

where the diagonal terms are the moments of inertia and the off-diagonal terms are the products of inertia. Inertia can be computed more directly by

$$J = \sum_{i=1}^N (\mathbf{x} - \mathbf{x}_c)(\mathbf{x} - \mathbf{x}_c)^T$$

The relationship between the inertia matrix and the symmetric ellipse matrix is

$$E = \frac{4}{m_{00}} J$$

To demonstrate this we can create a set of points that lie within the ellipse used in the example above

```

1  % generate a set of points within the ellipse
2  p = [];
3  while true
4      x = (rand(2,1)-0.5)*4;
5      if norm(x'*inv(E)*x) <= 1
6          p = [p x];
7      end
8      if numcols(p) >= 500
9          break;
10     end
11 end
12 plot(p(1,:), p(2,:), '.')
13
14 % compute the moments
15 m00 = mpq_point(p, 0,0);
16 m10 = mpq_point(p, 1,0);
17 m01 = mpq_point(p, 0,1);
18 xc = m10/m00; yc = m01/m00;
19
20 % compute second moments relative to centroid
21 pp = bsxfun(@minus, p, [xc; yc]);
22
23 m20 = mpq_point(pp, 2,0);
24 m02 = mpq_point(pp, 0,2);
25 m11 = mpq_point(pp, 1,1);
26
27 % compute the moments and ellipse matrix
28 J = [m20 m11; m11 m02];
29 E_est = 4 * J / m00

```

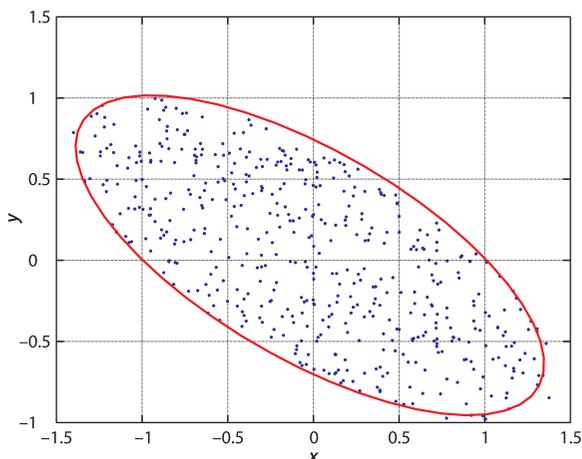


Fig. C.4. Data points (blue) with a fitted ellipse (red).

which results in an estimate

```
>> E_est
E_est =
    1.8706   -0.9151
   -0.9151    0.9716
```

which is similar to the original value of E . The point data is shown in Fig. C.4. We can overlay the estimated ellipse on the point data

```
>> plot_ellipse(E_est, [xc yc], 'r')
```

and the result is shown in red in Fig. C.4.

From a Set of Boundary Points

We wish to find the equation of an ellipse given a set of points (x_i, y_i) that define the boundary of an ellipse. Using the polynomial form of the ellipse Eq. C.5 for each point we write this in matrix form

$$\begin{pmatrix} y_1^2 & x_1 y_1 & x_1 & y_1 & 1 \\ y_2^2 & x_2 y_2 & x_2 & y_2 & 1 \\ & & \vdots & & \\ y_N^2 & x_N y_N & x_N & y_N & 1 \end{pmatrix} \begin{pmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \end{pmatrix} = \begin{pmatrix} -x_1^2 \\ -x_2^2 \\ \vdots \\ -x_N^2 \end{pmatrix}$$

and for $N \geq 5$ we can solve for the ellipse parameter vector.

C.2 Homogeneous Coordinates

A point in homogeneous coordinates, or the projective space \mathbb{P}^n , is represented by a coordinate vector $\tilde{x} = (\tilde{x}_1, \tilde{x}_2 \cdots \tilde{x}_{n+1})$. The Euclidean coordinates are related to the projective coordinates by

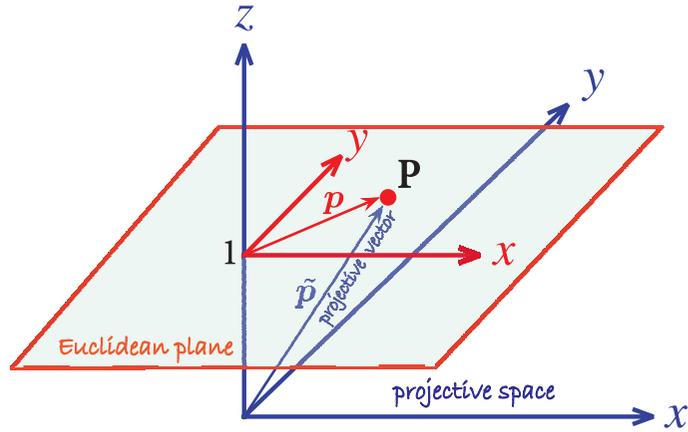
$$x_i = \frac{\tilde{x}_i}{\tilde{x}_{n+1}}, \quad i = 1 \cdots n$$

Conversely a homogeneous coordinate vector can be constructed from a Euclidean coordinate vector by

$$\tilde{x} = (x_1, x_2 \cdots x_n, 1)$$

and the tilde is used to indicate that the quantity is homogeneous.

Fig. C.5. A point P on the Euclidean plane \mathbb{R}^2 (red) is described by a coordinate vector $\mathbf{p} \in \mathbb{R}^2$ which is equivalent to the three-dimensional vector in the projective space \mathbb{P}^2 (blue) which is the homogeneous coordinate $\tilde{\mathbf{p}} \in \mathbb{P}^2$



The extra *degree of freedom* offered by projective coordinates has several advantages. It allows points and lines at infinity, known as ideal points and lines, to be represented using only real numbers. It also means that scale is unimportant, that is $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}' = \alpha \tilde{\mathbf{x}}$ both represent the same Euclidean point for all $\alpha \neq 0$. We express this as $\tilde{\mathbf{x}} \simeq \tilde{\mathbf{x}}'$. Points in homogeneous form can also be rotated with respect to a coordinate frame and translated simply by multiplying the homogeneous coordinate by an $(n + 1) \times (n + 1)$ homogeneous transformation matrix.

Homogeneous vectors are important in computer vision when we consider points and lines that exist in a plane – a camera’s image plane. We can also consider that the homogeneous form represents a ray in Euclidean space as shown in Fig. C.5. The relationship between points and rays is at the core of the projective transformation.

C.2.1 Two Dimensions

In two dimensions there is a duality between points and lines. In \mathbb{P}^2 a line is defined by a 3-tuple, $\tilde{\ell} = (\ell_1, \ell_2, \ell_3)^T$, not all zero, and the equation of the line is the set of all points

$$\tilde{\ell}^T \tilde{\mathbf{x}} = 0$$

which expands to $\ell_1 x + \ell_2 y + \ell_3 = 0$ and can be manipulated into the more familiar representation of a line. Note that this form can represent a vertical line, parallel to the y -axis, which the familiar form $y = mx + c$ cannot. This is the point equation of a line. The nonhomogeneous vector (ℓ_1, ℓ_2) is a normal to the line, and $(-\ell_2, \ell_1)$ is parallel to the line.

A point is defined by the intersection of two lines. If we write the point equations for two lines $\tilde{\ell}_1^T \tilde{\mathbf{p}} = 0$ and $\tilde{\ell}_2^T \tilde{\mathbf{p}} = 0$ their intersection is the point with coordinates

$$\tilde{\mathbf{p}} = \tilde{\ell}_1 \times \tilde{\ell}_2$$

and is known as the line equation of a point. Similarly, a line joining two points $\tilde{\mathbf{p}}_1$ and $\tilde{\mathbf{p}}_2$ is given by the cross-product

$$\tilde{\ell}_{12} = \tilde{\mathbf{p}}_1 \times \tilde{\mathbf{p}}_2$$

Consider the case of two parallel lines at 45° to the horizontal axis

```
>> l1 = [1 -1 0]';
>> l2 = [1 -1 -1]';
```

which we can plot

```
>> plot_homline(l1, 'b')
>> plot_homline(l2, 'r')
```

The intersection point of these parallel lines is

```
>> cross(l1, l2)
ans =
     1     1     0
```

This is an *ideal point* since the third coordinate is zero – the equivalent Euclidean point would be at infinity. Projective coordinates allow points and lines at infinity to be simply represented and manipulated without special logic.

The distance from a point with coordinates $\tilde{\mathbf{p}}$ to a line $\tilde{\ell}$ is

$$d = \frac{\tilde{\ell}^T \tilde{\mathbf{p}}}{p_3 \sqrt{\ell_1^2 + \ell_2^2}} \quad (\text{C.6})$$

C.2.1.1 Conics

Conic sections are an important family of planar curves that includes circles, ellipses, parabolas and hyperbolas which can be described by

$$Au^2 + Buv + Cv^2 + Du + Ev + F = 0$$

or more concisely as $\tilde{\mathbf{p}}^T \mathbf{c} \tilde{\mathbf{p}} = 0$ where \mathbf{c} is a matrix

$$\mathbf{c} = \begin{pmatrix} A & B/2 & D/2 \\ B/2 & C & E/2 \\ D/2 & E/2 & F \end{pmatrix}$$

The determinant of the top-left submatrix indicates the type of conic: negative for a hyperbola, 0 for a parabola and positive for an ellipse.

C.2.2 Three Dimensions

In three dimensions there is a duality between points and planes.

C.2.2.1 Lines

Using the homogeneous representation of the two points $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{q}}$ we can form a 4×4 skew-symmetric matrix

$$L = \tilde{\mathbf{q}}\tilde{\mathbf{p}}^T - \tilde{\mathbf{p}}\tilde{\mathbf{q}}^T = \begin{pmatrix} 0 & v_3 & -v_2 & -\omega_1 \\ -v_3 & 0 & v_1 & -\omega_2 \\ v_2 & -v_1 & 0 & -\omega_3 \\ \omega_1 & \omega_2 & \omega_3 & 0 \end{pmatrix}$$

whose 6 unique elements comprise the Plücker coordinate vector. This matrix is rank 2 and the determinant is a quadratic in the Plücker coordinates – a 4-dimensional quadric

hypersurface known as the Klein quadric. All points that lie on this manifold are valid lines. Many of the relationships in Sect. C.1.2.2 (between lines and points and planes) can be expressed in terms of this matrix. This matrix is returned by the `L` method of the `Plucker` class.

For a perspective camera with a camera matrix C the 3-dimensional Plücker line represented as a 4×4 skew-symmetric matrix L is projected onto the image plane as

$$\ell = CLC^T$$

which is a homogeneous line in \mathbb{P}^2 . This is computed automatically if a `Plucker` object is passed to the `project` method of a `CentralCamera` object.

C.2.2.2 Planes

The plane described by $\pi \tilde{x} = 0$ can be defined by a line and a point

$$\pi = L\tilde{p}$$

The join and incidence relationships are more complex than the cross products used for the 2-dimensional case. Three points define a plane and the join relationship is

$$\begin{pmatrix} \tilde{p}_1^T \\ \tilde{p}_2^T \\ \tilde{p}_3^T \end{pmatrix} \tilde{\pi} = 0$$

and the solution is found from the right-null space of the matrix. The incidence of three planes is the dual

$$\begin{pmatrix} \tilde{\pi}_1^T \\ \tilde{\pi}_2^T \\ \tilde{\pi}_3^T \end{pmatrix} \tilde{p} = 0$$

and is an ideal point, zero last component, if the planes do not intersect at a point.

C.2.2.3 Quadrics

Quadrics, short for quadratic surfaces, are a rich family of 3-dimensional surfaces. There are 17 standard types including spheres, ellipsoids, hyperboloids, paraboloids, cylinders and cones all described by

$$\tilde{x}^T Q \tilde{x} = 0$$

where $Q \in \mathbb{R}^{4 \times 4}$ is symmetric.

For a perspective camera with a camera matrix C the outline of the quadric is projected to the image plane by

$$c^* = CQ^*C^T$$

where c is a 3×3 matrix describing the conic, see Sect. C.2.1.1, and $(\cdot)^*$ represents the adjugate operation, see Appendix B.

C.3 Geometric Transformations

A linear transform is $\mathbf{y} = \mathbf{A}\mathbf{x}$ and an affine transform is

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b} \quad (\text{C.7})$$

which comprises a linear transformation *and* a change of origin. Examples of affine transformations include translation, scaling, homothety, similarity transformation, reflection, rotation, shear mapping, and compositions of them in any combination and sequence. Every linear transformation is affine, but not every affine transformation is linear.

In homogeneous coordinates we can write Eq. C.7 as

$$\tilde{\mathbf{y}} = \mathbf{H}\tilde{\mathbf{x}}, \text{ where } \mathbf{H} = \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & 1 \end{pmatrix}$$

and the transformation operates on a point with homogeneous coordinates $\tilde{\mathbf{x}}$. If a vector is defined as the difference between two homogeneous points $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{q}}$ then the difference $\tilde{\mathbf{p}} - \tilde{\mathbf{q}}$ is a 4-vector whose last element is zero, distinguishing a point from a vector.

Affine space is a generalization of Euclidean space and has no distinguished point that serves as an origin. Hence, no vector has a fixed origin and no vector can be uniquely associated to a point in an affine space, there are instead displacement vectors between two points of the space. Thus it makes sense to subtract two points of the space, giving a vector, but it does not make sense to add two points of the space. Likewise, it makes sense to add a vector to a point of an affine space, resulting in a new point displaced from the starting point by that vector.

In two-dimensions the most general transformation is projective transformation projective, also known as a collineation

$$\mathbf{H} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{pmatrix}$$

which is unique up to scale and one element has been normalized to one. It has 8 degrees of freedom.

The affine transformation is a subset where the elements of the last row are fixed

$$\mathbf{H} = \begin{pmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

and has 6 degrees of freedom.

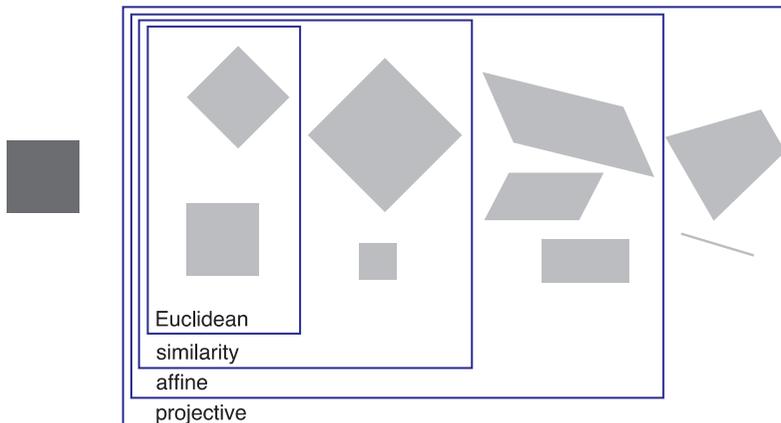


Fig. C.6. A 2-dimensional square (*dark grey*) is operated on by various transformations from the most limited (Euclidean) to the most general (projective)

Table C.1.
For various planar transformation families the possible geometric transformations and the geometric properties which are preserved are listed

	Euclidean	Similarity	Affine	Projective
Geometric transformation				
Rotation	✓	✓	✓	✓
Translation	✓	✓	✓	✓
Reflection		✓	✓	✓
Uniform scaling		✓	✓	✓
Nonuniform scaling			✓	✓
Shear			✓	✓
Perspective projection				✓
Preserved geometric properties (invariants)				
Length	✓			
Angle	✓	✓		
Ratio of lengths	✓	✓		
Parallelism	✓	✓	✓	
Incidence	✓	✓	✓	✓
Cross ratio	✓	✓	✓	✓

The similarity transformation is further subset

$$H = \begin{pmatrix} sR & t \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

where $R \in \mathbf{SO}(2)$ resulting in only 4 degrees of freedom. Similarity transforms, without reflection, are sometimes referred to as a Procrustes transform.

Finally the Euclidean or rigid-body transformation

$$H = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \in \mathbf{SE}(2)$$

is the most restrictive and has only 3 degrees of freedom. Some graphical examples of the effect of the various transformations on a square are shown in Fig. C.6. The possible geometric transformations for each type of transform are summarized in Table C.1 along with the geometric properties which are unchanged, or invariant, under that transformation. We see that while Euclidean is most restrictive in terms of the geometric transformations it can perform it is able to preserve important properties such as length and angle.

D

Lie Groups and Algebras

We cannot go very far in the study of rotations or rigid-body motion without coming across the terms Lie groups, Lie algebras or Lie brackets – all named in honor of the Norwegian mathematician Sophus Lie. Rotations and rigid-body motion in 2- and 3-dimensions can be represented by matrices which form Lie groups and which have Lie algebras.

We will start simply by considering the set of all real 2×2 matrices $A \in \mathbb{R}^{2 \times 2}$

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

which we could write as a linear combination of basis matrices

$$A = a_{11} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + a_{12} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + a_{21} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + a_{22} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

where each basis matrix represents a *direction* in a 4-dimensional space of 2×2 matrices. That is, the four axes of this space are *parallel* with each of these basis matrices. Any 2×2 matrix can be represented by a point in this space – this particular matrix is a point with the coordinates $(a_{11}, a_{12}, a_{21}, a_{22})$.

All proper rotation matrices, those belonging to $\text{SO}(2)$, are a subset of points within the space of all 2×2 matrices. For this example the points lie in a 1-dimensional subset, a closed curve, in the 4-dimensional space. This is an instance of a manifold, a lower-dimensional smooth *surface* embedded within a space.

The notion of a curve in the 4-dimensional space makes sense when we consider that the $\text{SO}(2)$ rotation matrix

$$A = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}$$

has only one free parameter, and varying that parameter moves the point along the manifold.



Sophus Lie (1842–1899) (surname pronounced lee) was a Norwegian mathematician who obtained his Ph.D. from the University of Christiania in Oslo in 1871. He spent time in Berlin working with Felix Klein, and later contributed to Klein's Erlangen program to characterize geometries based on group theory and projective geometry. On a visit to Milan during the Franco-Prussian war he was arrested as a German spy and spent one month in prison. He is best known for his discovery that continuous transformation groups (now called Lie groups) can be understood by linearizing them and studying their generating vector spaces. He is buried in the Vår Frelsers gravlund in Oslo. (Photograph by Ludwik Szacinski)

Invoking mathematical formalism we say that rotations $\text{SO}(2)$ and $\text{SO}(3)$, and rigid-body motions $\text{SE}(2)$ and $\text{SE}(3)$ are matrix Lie groups and this has two implications. Firstly, they are an *algebraic group*, a mathematical structure comprising elements and a single operator. In simple terms, a group \mathbf{G} has the following properties:

1. if g_1 and g_2 are elements of the group, that is $g_1, g_2 \in \mathbf{G}$, then the result of the group's operator \circ is also an element of the group: $g_1 \circ g_2 \in \mathbf{G}$. In general, groups are not commutative so $g_1 \circ g_2 \neq g_2 \circ g_1$. For rotations and rigid-body motions the group operator \circ represents composition. ▽
2. the group operator is associative, that is, $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$.
3. for $g \in \mathbf{G}$ there is an identity element $I \in \mathbf{G}$ such that $g \circ I = I \circ g = g$. ▽
4. for every $g \in \mathbf{G}$ there is a unique inverse $h \in \mathbf{G}$ such that $g \circ h = h \circ g = I$. ▶

In this book's notation the \oplus operator is the group operator.

In this book's notation the identity is denoted by 0 (implying null motion) so we can say that $\xi \oplus 0 = 0 \oplus \xi = \xi$.

In this book's notation we use the operator $\ominus \xi$ to form the inverse.

The second implication of being a Lie group is that there is a smooth (differentiable) manifold structure. At any point on the manifold we can construct tangent vectors. The set of all tangent vectors at that point form a vector space – the tangent space. This is the multidimensional equivalent to a tangent line on a curve, or a tangent plane on a solid. We can think of this as the set of all possible derivatives of the manifold at that point.

The tangent space *at the identity* is described by the Lie algebra of the group, and the basis directions of the tangent space are called the generators of the group. Points in this tangent space map to elements of the group via the exponential function. If \mathfrak{g} is the Lie algebra for group \mathbf{G} then

$$\forall X \in \mathfrak{g} \Rightarrow e^X \in \mathbf{G}$$

where the elements of \mathfrak{g} and \mathbf{G} are matrices of the same size and which each have a specific structure.

The surface of a sphere is a manifold in 3-dimensional space and at any point on that surface we can create a tangent vector. In fact we can create an infinite number of them and they lie within a plane which is a 2-dimensional vector space – the tangent space. We can choose a set of basis directions and establish a 2-dimensional coordinate system and we can map points on the plane to points on the sphere's surface.

Now consider an arbitrary real 3×3 matrix $A \in \mathbb{R}^{3 \times 3}$

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

which we could write as a linear combination of basis matrices

$$A = a_{11} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} + a_{12} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdots + a_{33} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where each basis matrix represents a *direction* in a 9-dimensional space of 3×3 matrices. Every possible 3×3 matrix is represented by a point in this space.

Not all matrices in this space are proper rotation matrices belonging to $\text{SO}(3)$, but those that do lie on a manifold since $\text{SO}(3)$ is a Lie group. The null rotation, represented by the identity matrix, is one point in this space. At that point we can construct a tangent space which has only 3 dimensions. Every point in the tangent space – the derivatives of the manifold – can be expressed as a linear combination of basis matrices

$$\Omega = \underbrace{\omega_1 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}}_{\mathbf{G}_1} + \underbrace{\omega_2 \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}}_{\mathbf{G}_2} + \underbrace{\omega_3 \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\mathbf{G}_3} \quad (\text{D.1})$$

The equivalent algebra is denoted using lower case letters and is a set of matrices.

which is the Lie algebra of the $\text{SO}(3)$ group. The bases of this space: \mathbf{G}_1 , \mathbf{G}_2 and \mathbf{G}_3 are called the generators of $\text{SO}(3)$ and belong to $\mathfrak{so}(3)$. ◀

Equation D.1 can be written as a skew-symmetric matrix parameterized by the vector $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3) \in \mathbb{R}^3$

$$\Omega = [\boldsymbol{\omega}]_{\times} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \in \mathfrak{so}(3)$$

and this reflects the 3 degrees of freedom of the $\text{SO}(3)$ group embedded in the space of all 3×3 matrices. The 3DOF is consistent with our intuition about rotations in 3D space and also Euler's rotation theorem.

Mapping between vectors and skew-symmetric matrices is frequently required and the following shorthand notation will be used

$$[\cdot]_{\times}: \mathbb{R} \mapsto \mathfrak{so}(2), \mathbb{R}^3 \mapsto \mathfrak{so}(3)$$

$$\vee_{\times}(\cdot): \mathfrak{so}(2) \mapsto \mathbb{R}, \mathfrak{so}(3) \mapsto \mathbb{R}^3$$

The first mapping is performed by the Toolbox function `skew` and the second by `vex` (which is named after the \vee_{\times}).

The exponential of *any* matrix in $\mathfrak{so}(3)$ is a valid member of $\text{SO}(3)$

$$\mathbf{R}(\theta, \hat{\boldsymbol{\omega}}) = e^{[\hat{\boldsymbol{\omega}}]_{\times} \theta} \in \text{SO}(3)$$

and an efficient closed-form solution is given by Rodrigues' rotation formula

$$\mathbf{R}(\theta, \hat{\boldsymbol{\omega}}) = \mathbf{I} + \sin \theta [\hat{\boldsymbol{\omega}}]_{\times} + (1 - \cos \theta) [\hat{\boldsymbol{\omega}}]_{\times}^2$$

Finally, consider an arbitrary real 4×4 matrix $\mathbf{A} \in \mathbb{R}^{4 \times 4}$

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

which we could write as a linear combination of basis matrices

$$\mathbf{A} = a_{11} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + a_{12} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \cdots + a_{44} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where each basis matrix represents a *direction* in a 16-dimensional space of all possible 4×4 matrices. Every 4×4 matrix is represented by a point in this space.

Not all matrices in this space are proper homogeneous transformation matrices belonging to $\text{SE}(3)$, but those that do lie on a smooth manifold. The null motion (zero rotation and translation), which is represented by the identity matrix, is one point in this space. At that point we can construct a tangent space, which has 6 dimensions in this case, and points in the tangent space can be expressed as a linear combination of basis matrices

$$\begin{aligned} \Sigma = & \omega_1 \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \omega_2 \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \omega_3 \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\ & + v_1 \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + v_2 \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + v_3 \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{aligned}$$

and these generator matrices belong to the Lie algebra of the group $\text{SE}(3)$ and are denoted $\mathfrak{se}(3)$. This can be written in general form as

$$\Sigma = [S] = \left(\begin{array}{ccc|c} 0 & -\omega_3 & \omega_2 & v_1 \\ \omega_3 & 0 & -\omega_1 & v_2 \\ -\omega_2 & \omega_1 & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{array} \right) \in \mathfrak{se}(3)$$

which is an augmented skew symmetric matrix parameterized by $S = (v, \omega) \in \mathbb{R}^6$ which is referred to as a twist and has physical interpretation in terms of a screw axis direction and position. The sparse matrix structure and this concise parameterization reflects the 6 degrees of freedom of the $\text{SE}(3)$ group embedded in the space of all 4×4 matrices. We extend our earlier shorthand notation

$$\begin{aligned} [\cdot]: \mathbb{R}^3 &\mapsto \mathfrak{se}(2), \mathbb{R}^6 \mapsto \mathfrak{se}(3) \\ \vee(\cdot): \mathfrak{se}(2) &\mapsto \mathbb{R}^3, \mathfrak{se}(3) \mapsto \mathbb{R}^6 \end{aligned}$$

We can use these operators to convert between a twist representation which is a 6-vector and a Lie algebra representation which is a 4×4 augmented skew-symmetric matrix. We convert the Lie algebra to the Lie group representation using

$$T(\theta, S) = e^{[S]\theta} \in \text{SE}(3)$$

or the inverse using the matrix logarithm. The exponential and the logarithm each have an efficient closed form solution.

Transforming a Twist – the Adjoint Representation

We have seen that rigid-body motions can be described by a twist which represents motion in terms of a screw axis direction and position, for example in Fig. D.1 the twist S_A can be used to transform points on the body. If the screw is rigidly attached to the body which undergoes some motion in $\text{SE}(3)$ the new twist is

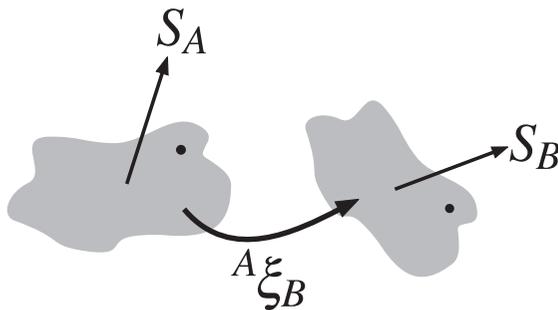
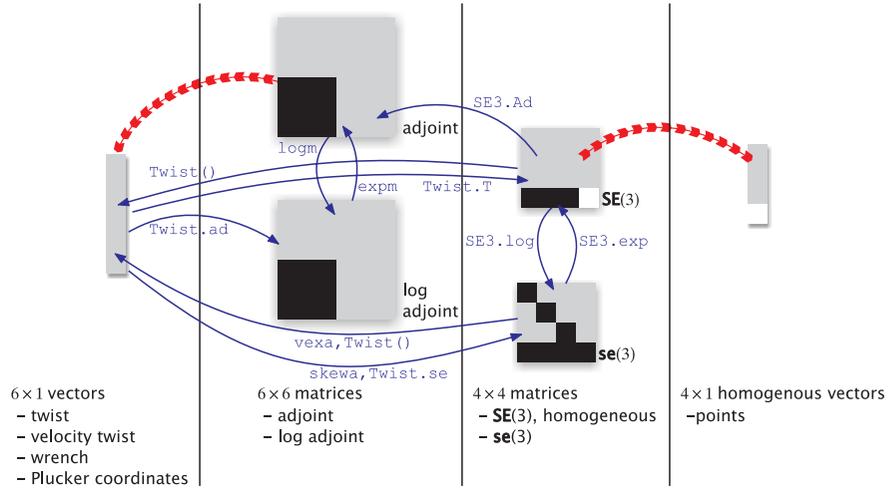


Fig. D.1. Points in the body (*grey cloud*) can be transformed by the twist S_A . If the body and the screw axis undergo a rigid-body transformation ${}^A\xi_B$ the new twist is S_B

Fig. D.2.
 The menagerie of $SE(3)$ related quantities. Matrix values are coded as: 0 (black), 1 (white), other values (grey). Transformations between types are indicated by blue arrows with the relevant class plus method name. Operations are indicated by red arrows: the tail-end object operates on the head-end object and results in another object of the head-end type



$${}^B S = \text{Ad}\left({}^B \xi_A\right) A S$$

where

$$\text{Ad}(\xi) = \begin{pmatrix} R & [t]_{\times} R \\ 0 & R \end{pmatrix} \in \mathbb{R}^{6 \times 6} \quad (\text{D.2})$$

is the adjoint representation of the rigid-body motion. Alternatively we can write

$$\text{Ad}\left(e^{[S]}\right) = e^{\text{ad}(S)}$$

where $\text{ad}(S)$ is the logarithm of the adjoint and defined in terms of the twist parameters as

$$\text{ad}(S) = \begin{pmatrix} [\omega]_{\times} & [v]_{\times} \\ 0 & [\omega]_{\times} \end{pmatrix} \in \mathbb{R}^{6 \times 6}$$

The relationship between the various mathematical objects discussed are shown in Fig. D.2.

E

Linearization, Jacobians and Hessians

In robotics and computer vision the equations we encounter are often nonlinear. To apply familiar and powerful analytic techniques we must work with linear or quadratic approximations to these equations. The principle is illustrated in Fig. E.1 for the 1-dimensional case, and the analytical approximations shown in red are made at $x = x_0$. The approximation equals the nonlinear function at x_0 but is increasing inaccurate as we move away from that point. This is called a local approximation since it is valid in a region local to x_0 – the size of the valid region depends on the severity of the nonlinearity. This approach can be extended to an arbitrary number of dimensions.

Scalar Function of a Scalar

The function $f: \mathbb{R} \mapsto \mathbb{R}$ can be expressed as a Taylor series

$$f(x_0 + \Delta) = f(x_0) + \frac{df}{dx}\Delta + \frac{1}{2}\frac{d^2f}{dx^2}\Delta^2 + \dots$$

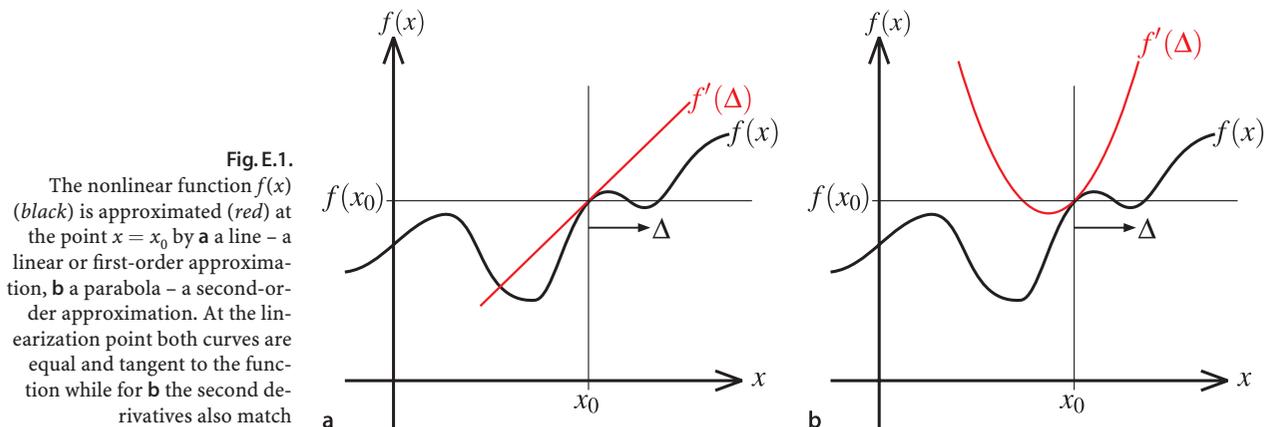
which we truncate to form a first-order or linear approximation

$$f'(\Delta) \approx f(x_0) + J(x_0)\Delta$$

or a second-order approximation

$$f'(\Delta) \approx f(x_0) + J(x_0)\Delta + \frac{1}{2}H(x_0)\Delta^2$$

where $\Delta \in \mathbb{R}$ is an infinitesimal change in x relative to the linearization point x_0 , and the first and second derivatives are given by $J(x_0) = df/dx|_{x_0}$ and $H(x_0) = d^2f/dx^2|_{x_0}$ respectively.



Scalar Function of a Vector

The scalar field $f(\mathbf{x}): \mathbb{R}^n \mapsto \mathbb{R}$ can be expressed as a Taylor series

$$f(\mathbf{x}_0 + \Delta) = f(\mathbf{x}_0) + J(\mathbf{x}_0)\Delta + \frac{1}{2}\Delta^T H(\mathbf{x}_0)\Delta + \dots$$

which we can truncate to form a first-order or linear approximation

$$f'(\Delta) \approx f(\mathbf{x}_0) + J(\mathbf{x}_0)\Delta$$

or a second-order approximation

$$f'(\Delta) \approx f(\mathbf{x}_0) + J(\mathbf{x}_0)\Delta + \frac{1}{2}\Delta^T H(\mathbf{x}_0)\Delta$$

where $\Delta \in \mathbb{R}^n$ is an infinitesimal change in $\mathbf{x} \in \mathbb{R}^n$ relative to the linearization point \mathbf{x}_0 , $J \in \mathbb{R}^{1 \times n}$ is the vector version of the first derivative, and $H \in \mathbb{R}^{n \times n}$ is the Hessian – the matrix version of the second derivative.

The derivative of the function $f(\cdot)$ with respect to the vector \mathbf{x} is

$$J(\mathbf{x}) = \nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

and is itself a vector that points in the direction at which the function $f(\mathbf{x})$ has maximal increase. It is often written as $\nabla_{\mathbf{x}} f$ to make explicit that the differentiation is with respect to \mathbf{x} .

The Hessian is an $n \times n$ symmetric matrix of second derivatives

$$H = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \frac{\partial^2 f}{\partial x_2 \partial x_n} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

The function is at a critical point when the Jacobian is not full rank. If the Hessian is positive definite then the function is at a local minimum, if negative definite then a local maximum, and if indefinite then the function is at a saddle point.

For functions which are quadratic in \mathbf{x} , as is the case for least-squares problems, it can be shown that the Hessian is

$$H(\mathbf{x}) = J(\mathbf{x})^T J(\mathbf{x}) + \sum_{i=1}^m f_i(\mathbf{x}) \frac{\partial^2 f_i}{\partial \mathbf{x}^2} \approx J(\mathbf{x})^T J(\mathbf{x})$$

which is frequently approximated by just the first term and this is key to Gauss-Newton least-squares optimization discussed in Sect. F.2.2.

Vector Function of a Vector

The vector field $\mathbf{f}(\mathbf{x}): \mathbb{R}^n \mapsto \mathbb{R}^m$ can be expressed as a Taylor series which can also be written as



Ludwig Otto Hesse (1811–1874) was a German mathematician, born in Königsberg, Prussia, who studied under Jacobi (p. 232) and Bessel at the University of Königsberg. He taught at Königsberg, Halle, Heidelberg and finally at the newly established Polytechnic School in Munich. In 1869 he joined the Bavarian Academy of Sciences.

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{pmatrix}$$

where $f_i: \mathbb{R}^m \rightarrow \mathbb{R}$ for $i \in \{1, 2, \dots, n\}$. The derivative of \mathbf{f} with respect to the vector \mathbf{x} can be expressed in matrix form as a Jacobian matrix

$$J = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

which can also be written as

$$J(\mathbf{x}) = \begin{pmatrix} \nabla f_1^T \\ \nabla f_2^T \\ \vdots \\ \nabla f_n^T \end{pmatrix}$$

This derivative is also known as the tangent map of \mathbf{f} , denoted $T\mathbf{f}$, or the differential of \mathbf{f} denoted $D\mathbf{f}$. To make explicit that the differentiation is with respect to \mathbf{x} this can be denoted as J_x , $T_x\mathbf{f}$, $D_x\mathbf{f}$ or even $\partial\mathbf{f}/\partial\mathbf{x}$.

The Hessian in this case is $\mathbf{H} \in \mathbb{R}^{n \times m \times n}$ which is a 3-dimensional array called a cubix.

Deriving Jacobians

Jacobians of functions are required for many optimization algorithms as well as for the extended Kalman filter, and can be evaluated numerically or symbolically.

Consider Eq. 6.8 for the range and bearing angle of a landmark given the pose of the vehicle and the position of the landmark. We can express this as the very simple MATLAB® anonymous function

```
>> zrange = @(xi, xv, w) ...
    [ sqrt((xi(1)-xv(1))^2 + (xi(2)-xv(2))^2) + w(1);
      atan((xi(2)-xv(2))/(xi(1)-xv(1)))-xv(3) + w(2) ];
```

To estimate the Jacobian $H_{xv} = \partial\mathbf{h}/\partial\mathbf{x}_v$ for $\mathbf{x}_v = (1, 2, \frac{\pi}{3})$ and $\mathbf{x}_i = (10, 8)$ we can compute a first-order numerical difference

```
>> xv = [1, 2, pi/3]; xi = [10, 8]; w = [0,0];
>> h0 = zrange(xi, xv, w)
h0 =
    10.8167
    -0.4592
>> d = 0.001;
>> J = [ zrange(xi, xv+[1,0,0]*d, w)-h0 ...
         zrange(xi, xv+[0,1,0]*d, w)-h0, ...
         zrange(xi, xv+[0,0,1]*d,w)-h0 ] / d
J =
   -0.8320   -0.5547         0
    0.0513   -0.0769   -1.0000
```

which shares the characteristic last column with the Jacobian shown in Eq. 6.14. Note that in computing this Jacobian we have set the measurement noise \mathbf{w} to zero. The principal difficulty with this approach is choosing \mathbf{d} , the difference used to compute the finite-difference approximation to the derivative. Too large and the results will be quite inaccurate if the function is nonlinear, too small and numerical problems will lead to reduced accuracy.

Alternatively we can perform the differentiation symbolically. This particular function is relatively simple and the derivatives can be determined easily using differential calculus. The numerical derivative can be used as a quick check for correctness. To avoid the possibility of error, or for more complex functions we can perform the differentiation symbolically using any of a large number of computer algebra packages. Using the MATLAB Symbolic Math Toolbox™ we can declare some symbolic variables

```
>> syms xi yi xv yv thetav wr wb
```

and then evaluate the same function as above

```
>> z = zrange([xi yi], [xv yv thetav], [wr wb])
z =
      wr + ((xi - xv)/(yi - yv)^2)^(1/2)
      wb - thetav + atan((yi - yv)/(xi - xv))
```

which is simply Eq. 6.8 in MATLAB symbolic form. The Jacobian is computed by a Symbolic Math Toolbox™ function

```
>> J = jacobian(z, [xv yv thetav])
J =
 [ -(2*xi - 2*xv)/(2*((xi - xv)^2 + (yi - yv)^2)^(1/2)), 0,
   -(2*yi - 2*yv)/(2*((xi - xv)^2 + (yi - yv)^2)^(1/2)), 0]
 [ (yi - yv)/((xi - xv)^2*((yi - yv)^2/(xi - xv)^2 + 1)),
   -1/((xi - xv)*((yi - yv)^2/(xi - xv)^2 + 1)), -1]
```

which has the required dimensions

```
>> about(J)
J [sym] : 2x3 (112 bytes)
```

and the characteristic last column. We could cut and paste this code into our program or automatically create a MATLAB callable function

```
>> Jf = matlabFunction(J);
```

where `Jf` is a MATLAB function handle. We can evaluate the Jacobian at the operating point given above

```
>> xv = [1, 2, pi/3]; xi = [10, 8]; w = [0,0];
>> Jf( xi(1), xv(1), xi(2), xv(2) )
ans =
   -0.8321   -0.5547         0
    0.0513   -0.0769   -1.0000
```

which is similar to the approximation above obtained numerically. The function `matlabFunction` can also write the function to an M-file. The functions `ccode` and `fcode` generate C and Fortran representations of the Jacobian.

Another interesting approach is the package ADOL-C which is an open-source tool for the automatic differentiation of C and C++ programs, that is, given a function written in C it will return a Jacobian function written in C. It is available at <http://www.coin-or.org/projects/ADOL-C.xml>.

F

Solving Systems of Equations

Solving systems of linear and nonlinear equations, particularly over-constrained systems, is a common problem in robotics and computer vision.

F.1 Linear Problems

F.1.1 Nonhomogeneous Systems

These are equations of the form

$$Ax = b$$

where we wish to solve for the unknown vector $x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ are constants.

If $n = m$ then A is square, and if A is nonsingular then the solution is obtained using the matrix inverse

$$x = A^{-1}b$$

In practice we often encounter systems where $m > n$, that is there are more equations than unknowns. In general there will not be an exact solution but we can attempt to find the *best* solution, in a least-squares sense, which is

$$x^* = \arg \min_x \|Ax - b\|$$

That solution is given by

$$x^* = (A^T A)^{-1} A^T b = A^+ b$$

Since the inverse left multiplies b .

which is known as the pseudo inverse or more formally the left-generalized inverse. Using SVD where $A = U \Sigma V^T$ this is

$$x = V \Sigma^{-1} U^T b$$

where Σ^{-1} is simply the element-wise inverse of the diagonal elements of Σ^T .

If the matrix is singular, or the system is under constrained $n < m$, then there are infinitely many solutions. We can again use the SVD approach

$$x = V \Sigma^{-1} U^T b$$

where this time Σ^{-1} is the element-wise inverse of the *nonzero* diagonal elements of Σ , all other zeros are left in place.

In MATLAB all these problems can be solved using the backslash operator

```
>> x = A\b
```

For the problem

$$RP = Q$$

where R is an unknown rotation matrix in $\text{SO}(n)$, and $P = \{p_1 \cdots p_m\} \in \mathbb{R}^{n \times m}$ and $Q = \{q_1 \cdots q_m\} \in \mathbb{R}^{n \times m}$ comprise column vectors for which $q_i = Rp_i$. We first compute the moment matrix

$$M = \sum_{i=1}^N q_i p_i^T$$

and then compute the SVD $M = U\Sigma V^T$. The least squares estimate of the rotation matrix is

$$R = UV^T$$

and is guaranteed to be an orthogonal matrix.

F.1.2 Homogeneous Systems

These are equations of the form

$$Ax = 0$$

and always have the trivial solution $x = 0$. If A is square and nonsingular this is the only solution. Otherwise, if A is not of full rank, that is the matrix is nonsquare, or square and singular then there are an infinite number of solutions which are linear combinations of vectors in the right null space of A which is computed by the MATLAB function `null`.

F.2 Nonlinear Problems

Many problems in robotics and computer vision involves sets of nonlinear equations. Solution of these problems requires linearizing the equations about an estimated solution, solving for an improved solution and iterating. Linearization is discussed in Appendix E.

F.2.1 Finding Roots

Consider a set of equations expressed in the form

$$f(x) = 0$$

where $f: \mathbb{R}^n \mapsto \mathbb{R}^m$. This is a nonlinear version of the homogeneous system described above. We first linearize the equation about our best estimate of the solution x_0

$$f(x_0 + \Delta) = f(x_0) + J(x_0)\Delta + \frac{1}{2}\Delta^T H(x_0)\Delta + \cdots \quad (\text{F.1})$$

where $\Delta \in \mathbb{R}^n$ is an infinitesimal change in x relative to x_0 . We truncate this to form a linear approximation

$$f'(\Delta) \approx f_0 + J\Delta \quad (\text{F.2})$$

where $f_0 = f(x_0)$ is the function value and $J = J(x_0) \in \mathbb{R}^{1 \times n}$ the Jacobian, both evaluated at the linearization point. Now we solve an approximation of our original problem $f'(\Delta) = 0$

$$f_0 + J\Delta = 0 \Rightarrow \Delta = -J^{-1}f_0$$

If $n \neq m$ then J is nonsquare and we can use the pseudo-inverse or the MATLAB backslash operator `-J \ f0`. The computed step Δ is based on an approximation to the original nonlinear function so $x_0 + \Delta$ will generally not be the solution but it will be closer. This leads to an iterative solution – the Newton-Raphson method:

```

1  while  $\|f(x_0)\| > \varepsilon$  do
2      compute  $f_0 = f(x_0), J(x_0)$ 
3       $\Delta = -J^{-1}f_0$ 
4       $x_0 \leftarrow x_0 + \Delta$ 
5  end
```

F.2.2 Nonlinear Minimization

A very common class of problems involves finding the *minimum* of a scalar function $f(x): \mathbb{R}^n \mapsto \mathbb{R}$ which can be expressed as

$$x^* = \underset{x}{\operatorname{arg\,min}} f(x)$$

The derivative of the linearized system Eq. F.2 is

$$\frac{df'}{d\Delta} = J$$

and if we consider the function to be a multi-dimensional surface then $J(x_0)$ is vector indicating the direction and magnitude of the *slope* at $x = x_0$ so an update of

$$\Delta = -\beta J$$

will move the estimate *down hill* toward the minimum. This leads to an iterative solution called gradient descent:

```

1  repeat
2      compute  $J = J(x_0)$ 
3       $\Delta = -\beta J$ 
4       $x_0 \leftarrow x_0 + \Delta$ 
5  until  $\|\Delta\| < \varepsilon$ 
```

and the challenge is to choose the appropriate step size β .

If we include the second-order term from Eq. F.1 the approximation becomes

$$f'(\Delta) \approx f_0 + J\Delta + \frac{1}{2}\Delta^T H(x_0)\Delta$$

and to find its minima we take the derivative and set it to zero

$$\frac{df'}{d\Delta} = 0 \Rightarrow J + H\Delta = 0$$

and the update is

$$\Delta = -H^{-1}J$$

This leads to another iterative solution – Newton’s method. The challenge is determining the Hessian of the nonlinear system, either by numerical approximation or symbolic manipulation.

F.2.3 Nonlinear Least Squares Minimization

Very commonly the scalar function we wish to optimize is a quadratic cost function

$$F(x) = \|\mathbf{f}(x)\|^2 = \mathbf{f}(x)^T \mathbf{f}(x)$$

where $\mathbf{f}(x): \mathbb{R}^n \mapsto \mathbb{R}^m$ is some vector-valued nonlinear function which we can linearize as

$$\mathbf{f}'(\Delta) \approx \mathbf{f}_0 + J\Delta$$

and the scalar cost is

$$\begin{aligned} F(\Delta) &\approx (\mathbf{f}_0 + J\Delta)^T (\mathbf{f}_0 + J\Delta) \\ &\approx \mathbf{f}_0^T \mathbf{f}_0 + \boxed{\mathbf{f}_0^T J\Delta + \Delta^T J^T \mathbf{f}_0} + \Delta^T J^T J\Delta \\ &\approx \mathbf{f}_0^T \mathbf{f}_0 + 2\mathbf{f}_0^T J\Delta + \Delta^T J^T J\Delta \end{aligned}$$

One term is the transpose of the other, but since both result in a scalar transposition doesn't matter.

where $J^T J \in \mathbb{R}^{n \times n}$ is the *approximate* Hessian from page 618.

To minimize the error of this linearized least squares system we take the derivative with respect to Δ and set it to zero

$$\frac{dF}{d\Delta} = 0 \Rightarrow 2\mathbf{f}_0^T J + 2J^T J\Delta = 0$$

which we can solve for the locally optimal update

$$\Delta = -(J^T J)^{-1} J^T \mathbf{f}_0 \quad (\text{F.3})$$

where we can recognize the pseudo or left generalized-inverse of J . Once again we iterate to find the solution – a Gauss-Newton iteration.

Numerical Issues

When solving Eq. F.3 we may find that the Hessian $J^T J$ is poorly conditioned or singular and this can be remedied by adding a damping term

$$\Delta = -(J^T J + \lambda I)^{-1} J^T \mathbf{f}_0$$

which makes the system more positive definite. Since $J^T J + \lambda I$ is effectively in the denominator, increasing λ will decrease $\|\Delta\|$ and slow convergence.

How do we choose λ ? We can experiment with different values but a better way is the Levenberg-Marquardt algorithm (Algorithm F.1) which adjusts λ to ensure convergence. If the error increases compared to the last step then the step is repeated with increased λ to reduce the step size. If the error decreases then λ is reduced to increase the convergence rate. The updates vary continuously between Gauss-Newton (low λ) and gradient descent (high λ).

For problems where n is large inverting the $n \times n$ approximate Hessian is expensive. Typically $m < n$ which means the Jacobian is not square and Eq. F.3 can be rewritten as

Algorithm F.1.
Levenberg-Marquardt algorithm, c is typically chosen in the range 2 to 10

```

1 initialize  $\lambda$ 
2 repeat
3   compute  $\mathbf{f}_0 = \mathbf{f}(\mathbf{x}_0), \mathbf{J} = \mathbf{J}(\mathbf{x}_0), \mathbf{H} = \mathbf{J}^T \mathbf{J}$ 
4    $\Delta = -(\mathbf{H} + \lambda \mathbf{I})^{-1} \mathbf{J}^T \mathbf{f}_0$ 
5   if  $F(\mathbf{x}_0 + \Delta) < F(\mathbf{x}_0)$  then
6     - error decreased: reduce damping
7      $\mathbf{x}_0 \leftarrow \mathbf{x}_0 + \Delta$ 
8      $\lambda \leftarrow \lambda / c$ 
9   else
10    - error increased: discard and raise damping
11     $\lambda \leftarrow c\lambda$ 
12  end
13 until  $\|\Delta\| < \varepsilon$ 

```

$$\Delta = -\mathbf{J}^T (\mathbf{J}\mathbf{J}^T)^{-1} \mathbf{f}_0$$

which is the right pseudo-inverse and involves inverting a smaller matrix. We can reintroduce a damping term

$$\Delta = -\mathbf{J}^T (\mathbf{J}\mathbf{J}^T + \lambda \mathbf{I})^{-1} \mathbf{f}_0$$

and if λ is large this becomes simply

$$\Delta \approx -\beta \mathbf{J}^T \mathbf{f}_0$$

but exhibits very slow convergence.

If $\mathbf{f}_k(\cdot)$ has additive noise that is zero mean, normally distributed and time invariant we have a maximum likelihood estimator of \mathbf{x} . Outlier data has a significant impact on the result since errors are squared. Robust estimators minimize the effect of outlier data and in an M-estimator

$$F(\mathbf{x}) = \rho(\mathbf{f}_k(\mathbf{x}))$$

the squared norm is replaced by a loss function $\rho(\cdot)$ which models the likelihood of its argument. Unlike the squared norm these functions flatten off for large values, and some common examples include the Huber loss function and the Tukey biweight function.

F.2.4 Sparse Nonlinear Least Squares

For a large class of problems the overall cost is the sum of quadratic costs

$$F(\mathbf{x}) = \sum_k \|\mathbf{f}_k(\mathbf{x})\|^2 = \sum_k \mathbf{f}_k(\mathbf{x})^T \mathbf{f}_k(\mathbf{x}) \quad (\text{F.4})$$

Consider the problem of fitting a model $\mathbf{z} = \phi(\mathbf{w}; \mathbf{x})$ where $\phi: \mathbb{R}^p \mapsto \mathbb{R}^m$ with parameters $\mathbf{x} \in \mathbb{R}^n$ to a set of data points $(\mathbf{w}_k, \mathbf{z}_k)$. The error vector associated with the k^{th} data point is

$$\mathbf{f}_k(\mathbf{x}) = \mathbf{z}_k - \phi(\mathbf{w}_k; \mathbf{x}) \in \mathbb{R}^m$$

and minimizing Eq. F.4 gives the optimal model parameters \mathbf{x} .

Another example is pose-graph optimization as used for pose-graph SLAM and bundle adjustment. Edge k in the graph connects vertices i and j and has an associated cost $f_k(\cdot): \mathbb{R}^n \mapsto \mathbb{R}^m$

$$f_k(\mathbf{x}) = \hat{e}_k(\mathbf{x}) - e_k^\# \quad (\text{F.5})$$

where $e_k^\#$ is the observed value of the edge parameter and $\hat{e}_k(\mathbf{x})$ is the estimate based on the state \mathbf{x} of the pose graph. This is linearized

$$f_k'(\Delta) \approx f_{0,k} + J_k \Delta$$

and the squared error for the edge is

$$F_k(\mathbf{x}) = f_k(\mathbf{x})^T \Omega_k f_k(\mathbf{x})$$

where $\Omega_k \in \mathbb{R}^{m \times m}$ is a positive-definite constant matrix \blacktriangleright which we combine as

$$\begin{aligned} F_k(\Delta) &\approx (f_{0,k} + J_k \Delta)^T \Omega_k (f_{0,k} + J_k \Delta) \\ &\approx f_{0,k}^T \Omega_k f_{0,k} + f_{0,k}^T \Omega_k J_k \Delta + \Delta^T J_k^T \Omega_k f_{0,k} + \Delta^T J_k^T \Omega_k J_k \Delta \\ &\approx c_k + 2\mathbf{b}_k^T \Delta + \Delta^T \mathbf{H}_k \Delta \end{aligned}$$

This can be used to specify the significance of the edge $\det \Omega_k$ with respect to other edges, as well as the relative significance of the elements of $f_k(\cdot)$.

where $\mathbf{b}_k^T = f_{0,k}^T \Omega_k J_k$ and $\mathbf{H}_k = \sum_k J_k^T \Omega_k J_k$. The total cost is the sum of all edge costs

$$\begin{aligned} F(\Delta) &= \sum_k F_k(\Delta) \\ &\approx \sum_k (c_k + 2\mathbf{b}_k^T \Delta + \Delta^T \mathbf{H}_k \Delta) \\ &\approx \sum_k c_k + 2 \left(\sum_k \mathbf{b}_k^T \right) \Delta + \Delta^T \left(\sum_k \mathbf{H}_k \right) \Delta \\ &\approx c + 2\mathbf{b}^T \Delta + \Delta^T \mathbf{H} \Delta \end{aligned}$$

where $\mathbf{b}^T = \sum_k f_{0,k}^T \Omega_k J_k$ and $\mathbf{H} = \sum_k J_k^T \Omega_k J_k$ are summations over the edges of the graph. Once they are computed we proceed as previously, taking the derivative with respect to Δ and setting it to zero, solving for the update Δ and iterating using Algorithm F.1.

State Vector

The state vector is a concatenation of all poses and coordinates in the optimization problem. For pose-graph SLAM it takes the form

$$\mathbf{x} = \{\xi_1, \xi_2 \dots \xi_N\} \in \mathbb{R}^n$$

Poses must be represented in a vector form and preferably one that is compact and singularity free. For $\text{SE}(2)$ this is quite straightforward and we use $\xi \sim (x, y, \theta) \in \mathbb{R}^3$. For $\text{SE}(3)$ we will use $\xi \sim (\mathbf{t}, \mathbf{r}) \in \mathbb{R}^6$ which comprises translation $\mathbf{t} \in \mathbb{R}^3$ and rotation $\mathbf{r} \in \mathbb{R}^3$. The latter can be triple angles (Euler or roll-pitch-yaw), axis-angle, exponential coordinates or the vector part of a unit-quaternion as discussed on page 499. The state vector has structure, comprising a sequence of subvectors one per pose. We denote the i^{th} subvector of \mathbf{x} as $\mathbf{x}_i \in \mathbb{R}^{N_\xi}$, where $N_\xi = 3$ for $\text{SE}(2)$ and $N_\xi = 6$ for $\text{SE}(3)$.

For pose-graph SLAM with landmarks, or bundle adjustment the state vector comprises poses and coordinate vectors

$$\mathbf{x} = \{\xi_1, \xi_2 \cdots \xi_N | \mathbf{P}_1, \mathbf{P}_2 \cdots \mathbf{P}_M\} \in \mathbb{R}^n$$

and the i^{th} and j^{th} subvectors of \mathbf{x} are denoted $\mathbf{x}_i \in \mathbb{R}^{N\xi}$ and $\mathbf{x}_j \in \mathbb{R}^{Np}$ and correspond to ξ_i and \mathbf{P}_j respectively.

Inherent Structure

A key observation is that the error vector $\mathbf{f}_k(\mathbf{x})$ for edge k depends only on the associated vertices i and j , and this means that the Jacobian

$$J_k = \frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}} \in \mathbb{R}^{m \times m}$$

is mostly zeros

$$J_k = (0 \cdots A_i \cdots B_j \cdots 0), \quad A_i = \frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}_i}, \quad B_j = \frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}_j}$$

where $A_i \in \mathbb{R}^{m \times N\xi}$ and $B_j \in \mathbb{R}^{m \times N\xi}$ or $B_j \in \mathbb{R}^{m \times Np}$ according to the state vector structure.

This sparse block structure means that the vector \mathbf{b}_k and the Hessian $J_k^T \Omega_k J_k$ also have a sparse block structure as shown in Fig. F.1. The Hessian has just four small nonzero blocks so rather than compute the product $J_k^T \Omega_k J_k$, which involves many multiplications by zero, we can just compute the four nonzero blocks and add them into the Hessian for the least squares system. All blocks in a row have the same height, and in a column have the same width. For pose-graph SLAM with landmarks, or bundle adjustment the blocks are of different sizes as shown in Fig. F.1b.

If the value of an edge represents pose then Eq. F.5 must be replaced with $\mathbf{f}_k(\mathbf{x}) = \hat{\mathbf{e}}_k(\mathbf{x}) \ominus \mathbf{e}_k^\#$. We generalize this with the \boxminus operator to indicate that the use of $-$ or \ominus as appropriate. Similarly when updating the state vector at the end of an iteration the poses must be compounded $\mathbf{x}_0 \leftarrow \mathbf{x}_0 \oplus \Delta$ and we generalize this to the \boxplus operator. The pose-graph optimization is solved by the iteration in Algorithm F.2.

```

1  repeat
2       $H \leftarrow 0, \mathbf{b} \leftarrow 0$ 
3      for each  $k$  do
4           $\mathbf{f}_{0,k}(\mathbf{x}_0) = \hat{\mathbf{e}}_k(\mathbf{x}) \boxminus \mathbf{e}_k^\#$ 
5           $(i, j) = \text{vertices}(k)$ 
6          compute  $A_i(\mathbf{x}_i), B_j(\mathbf{x}_j)$ 
7           $\mathbf{b}_i \leftarrow \mathbf{b}_i + \mathbf{f}_{0,k}^T \Omega_k A_i$ 
8           $\mathbf{b}_j \leftarrow \mathbf{b}_j + \mathbf{f}_{0,k}^T \Omega_k B_j$ 
9           $H_{i,i} \leftarrow H_{i,i} + A_i^T \Omega_k A_i$ 
10          $H_{i,j} \leftarrow H_{i,j} + A_i^T \Omega_k B_j$ 
11          $H_{j,i} \leftarrow H_{j,i} + B_j^T \Omega_k A_i$ 
12          $H_{j,j} \leftarrow H_{j,j} + B_j^T \Omega_k B_j$ 
13     end
14      $\Delta = -H^{-1} \mathbf{b}$ 
15      $\mathbf{x}_0 \leftarrow \mathbf{x}_0 \boxplus \Delta$ 
16 until  $\|\Delta\| < \varepsilon$ 
    
```

Algorithm F.2.

Pose graph optimization. For Levenberg-Marquardt optimization replace line 14 with lines 4–12 from Algorithm F.1

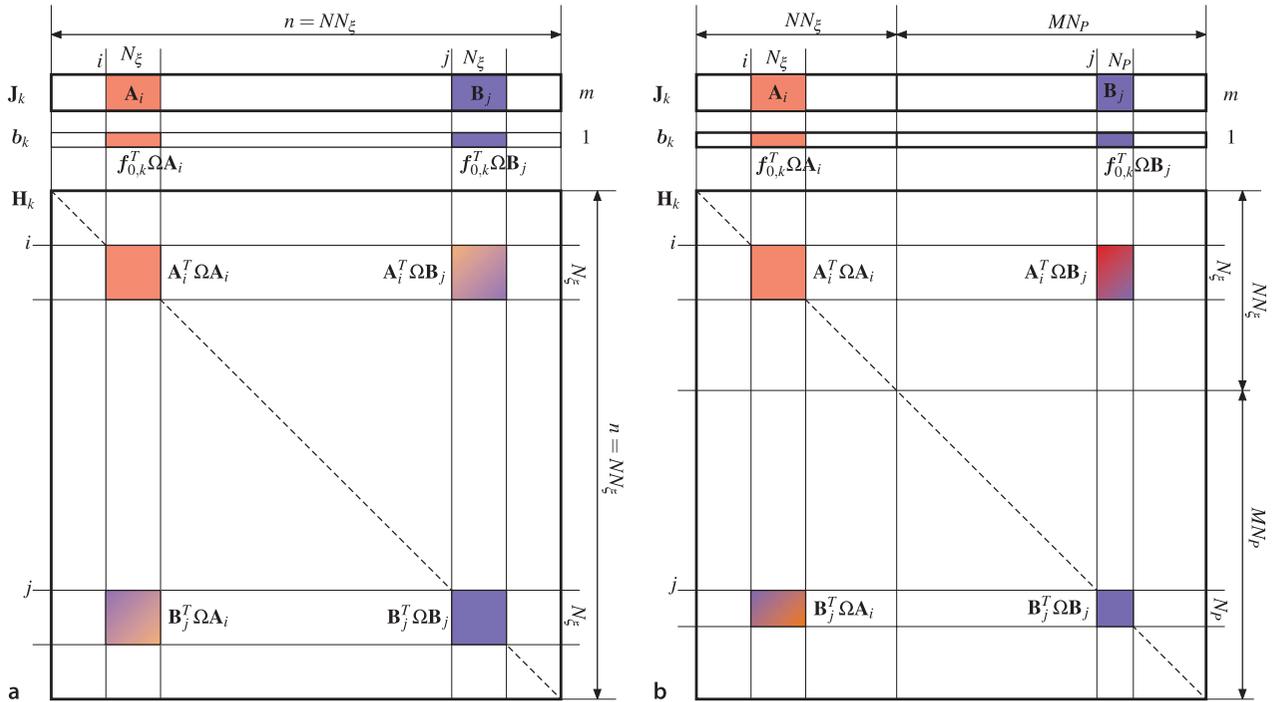


Fig. F.1. Inherent structure of the error vector, Jacobian and Hessian matrices for graph-based least-squares problems. **a** Pose-graph SLAM with N nodes representing robot pose as $\mathbb{R}^{N\xi}$; **b** bundle adjustment with N nodes representing camera pose as $\mathbb{R}^{N\xi}$ and M nodes representing landmark position as \mathbb{R}^{Np} . The indices i and j denote the i^{th} and j^{th} block not the i^{th} and j^{th} row or column. *White* indicates zero values

Large Scale Problems

For pose-graph SLAM with thousands of poses or bundle adjustment with thousands of cameras and millions of landmarks the Hessian matrix will be massive leading to computation and storage challenges. The overall Hessian is the summation of many edge Hessians structured as shown in Fig. F.1 and the total Hessian for two problems we have discussed are shown in Fig. F.2. They have clear structure which we can exploit.

Firstly, in both cases the Hessian is sparse – that is, it contains mostly zeros. MATLAB has built-in support for such matrices and instead of storing all those zeros (at 8 bytes each) it simply keeps a list of the nonzero elements. All the standard matrix operations employ efficient algorithms for manipulating sparse matrices.

Secondly, for the bundle adjustment case we see that the Hessian has two block diagonal submatrices so we partition the system as

$$\begin{pmatrix} \mathbf{B} & \mathbf{E} \\ \mathbf{E}^T & \mathbf{C} \end{pmatrix} \begin{pmatrix} \Delta_\xi \\ \Delta_p \end{pmatrix} = \begin{pmatrix} \mathbf{b}_\xi \\ \mathbf{b}_p \end{pmatrix}$$

where \mathbf{B} and \mathbf{C} are block diagonal. The subscripts ξ and P denote the blocks of Δ and \mathbf{b} associated with camera poses and landmark positions respectively. We solve first for the camera pose updates Δ_ξ

$$\mathbf{S}\Delta_\xi = \mathbf{b}_\xi - \mathbf{E}\mathbf{C}^{-1}\mathbf{b}_p$$

where $\mathbf{S} = \mathbf{B} - \mathbf{E}\mathbf{C}^{-1}\mathbf{E}^T$ is the Schur complement which is a symmetric positive-definite matrix that is also block diagonal. Then we solve for the update to landmark positions

$$\Delta_p = \mathbf{C}^{-1}(\mathbf{b}_p + \mathbf{E}^T\Delta_\xi)$$

More sophisticated techniques exploit the fine-scale block structure to further reduce computational time, for example GTSAM (<https://bitbucket.org/gtborg/gtsam>) and SLAM++ (<https://sourceforge.net/projects/slam-plus-plus>).

A block diagonal matrix is inverted by simply inverting each of the nonzero blocks along the diagonal.

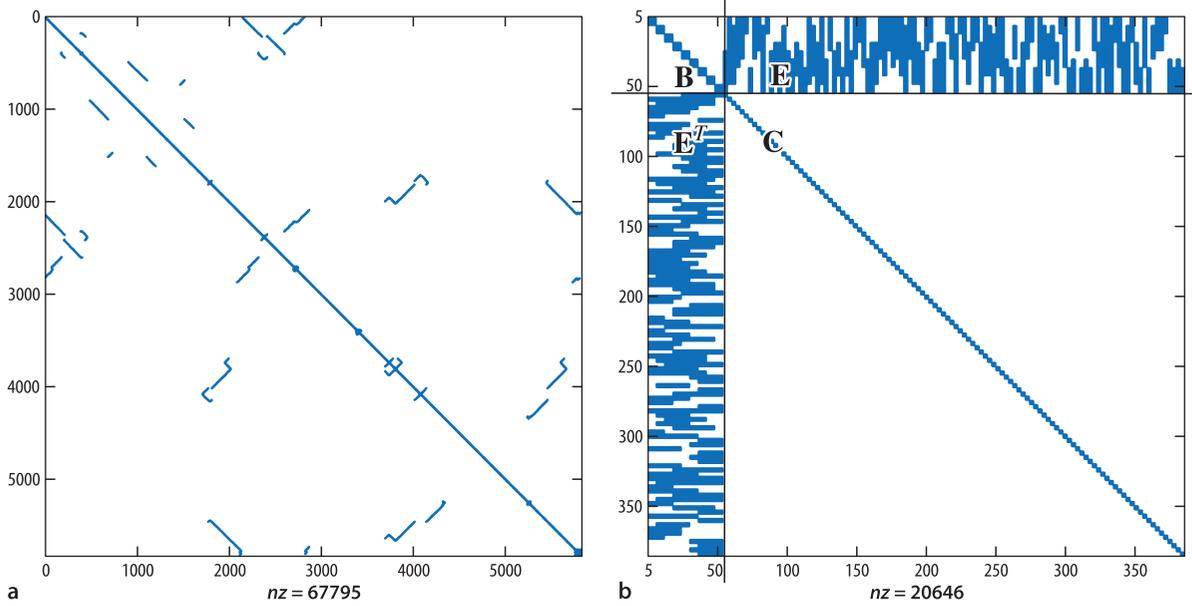


Fig. F.2. Hessian sparsity maps produced using the MATLAB `spy` function, the number of non-zero elements is shown beneath the plot. **a** Hessian for the pose-graph SLAM problem of Fig. 6.17, the diagonal elements represent pose constraints between successive nodes due to odometry, the off-diagonal terms represent constraints due to revisiting locations (loop closures); **b** Hessian for a bundle adjustment problem with 10 cameras and 110 landmarks (vision/examples/bademo.m)

Anchoring

Optimization provides a solution where the *relative* poses and positions give the lowest overall cost, and the solution will have an arbitrary transformation with respect to a global reference frame. To obtain absolute poses and positions we must anchor or fix some nodes – assign them values with respect to the global frame and prevent the optimization from adjusting them. The appropriate way to achieve this is to remove from H and b the rows and columns corresponding to the anchored poses and positions. We then solve a lower dimensional problem for Δ' which will be shorter than x and careful book keeping is required to correctly match the subvectors of Δ' with those of x for the update.

G

Gaussian Random Variables

The 1-dimensional Gaussian function

$$g(x) = \frac{1}{\sqrt{\sigma^2 2\pi}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \quad (\text{G.1})$$

is described by the position of its peak μ and its width σ . The total area under the curve is unity and $g(x) > 0, \forall x$. The function can be plotted using the Toolbox function `gaussfunc`

```
>> x = linspace(-6, 6, 500);
>> plot(x, gaussfunc(0, 1, x), 'r' )
>> hold on
>> plot(x, gaussfunc(0, 2^2, x), '--b' )
```

and Fig. G.1 shows two Gaussians with zero mean and $\sigma = 1$ and $\sigma = 2$. Note that the second argument to `gaussfunc` is the variance not standard deviation.

If the Gaussian is considered to be a probability density function (PDF) then this is the well known normal distribution and the peak position μ is the mean value and the width σ is the standard deviation. A random variable drawn from a normal distribution is often written as $X \sim N(\mu, \sigma^2)$, and $N(0, 1)$ is referred to as the standard normal distribution – the MATLAB function `randn` draws random numbers from this distribution. To draw one hundred Gaussian random numbers with mean `mu` and standard deviation `sigma` is

```
>> g = sigma * randn(100) + mu;
```

The probability that a random value falls within an interval $x \in [x_1, x_2]$ is obtained by integration

$$P = \int_{x_1}^{x_2} g(x)dx = \Phi(x_2) - \Phi(x_1) \quad \text{where} \quad \Phi(x) = \int_{-\infty}^x g(x)dx$$

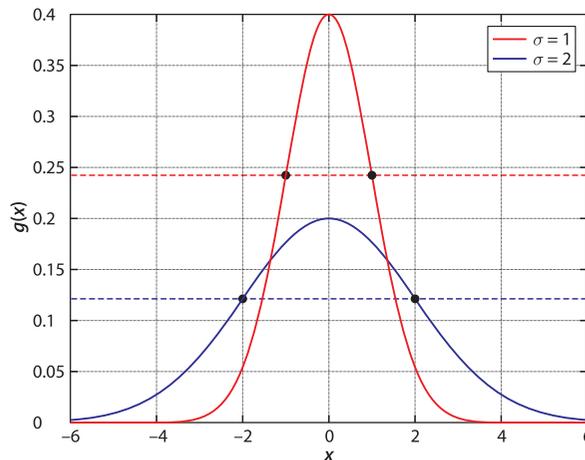


Fig. G.1.

Two Gaussian functions, both with mean $\mu = 0$, and with standard deviation $\sigma = 1$, and $\sigma = 2$. The markers indicate the points $x = \mu \pm 1\sigma$. The blue curve is wider but less tall, since the total area under the curve is unity

or evaluation of the cumulative normal distribution function $\Phi(x)$. The marked points in Fig. G.1 at $\mu \pm 1\sigma$ delimit the 1σ confidence interval. The area under the curve over this interval is 0.68, so the probability of a random value being drawn from this interval is 68%.

The Gaussian can be extended to an arbitrary number of dimensions. The n -dimensional Gaussian, or multivariate normal distribution, is

$$g(\mathbf{x}) = \frac{1}{\sqrt{\det(\mathbf{P})(2\pi)^n}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{P}^{-1}(\mathbf{x}-\boldsymbol{\mu})} \quad (\text{G.2})$$

and compared to the scalar case of Eq. G.1 $\mathbf{x} \in \mathbb{R}^n$ and $\boldsymbol{\mu} \in \mathbb{R}^n$ have become vectors, the squared term in the exponent has been replaced by a matrix quadratic form, and σ^2 , the variance, has become a positive-definite (and hence symmetric) covariance matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$. The diagonal elements represent the variance of x_i and the off-diagonal elements P_{ij} are the correlations between x_i and x_j . If the variables are independent or uncorrelated the matrix \mathbf{P} would be diagonal. The covariance matrix is symmetric and positive definite.

We can plot a 2-dimensional Gaussian

```
>> [x,y] = meshgrid(-5:0.1:5, -5:0.1:5);
>> P = diag([1 2^2]);
>> surf(x, y, gaussfunc([0 0], P, x, y))
```

as a surface which is shown in Fig. G.2. In this case $\boldsymbol{\mu} = (0, 0)$ and $\mathbf{P} = \text{diag}(1^2, 2^2)$ which corresponds to uncorrelated variables with standard deviation of 1 and 2 respectively. Figure G.2 also shows a number of elliptical contours – contours of constant probability density. If this 2-dimensional probability density function represents the position of a robot in the xy -plane the most likely position for the robot is at $(0, 0)$ and the size of the ellipse says something about our spatial certainty. A particular contour indicates the boundary of a region within which the robot is located with a particular probability. A large ellipse indicates we know, with that probability, that the robot is somewhere inside a large area – we have low certainty about the robot's position. Conversely, a small ellipse means that we know the robot, with the same probability, is somewhere within a much smaller area.

The contour lines are ellipses and in this example the radii in the y - and x -directions are in the ratio 2:1 as defined by the ratio of the standard deviations. For higher order Gaussians, $n > 2$, the corresponding confidence interval is the surface of an ellipsoid in n -dimensional space.

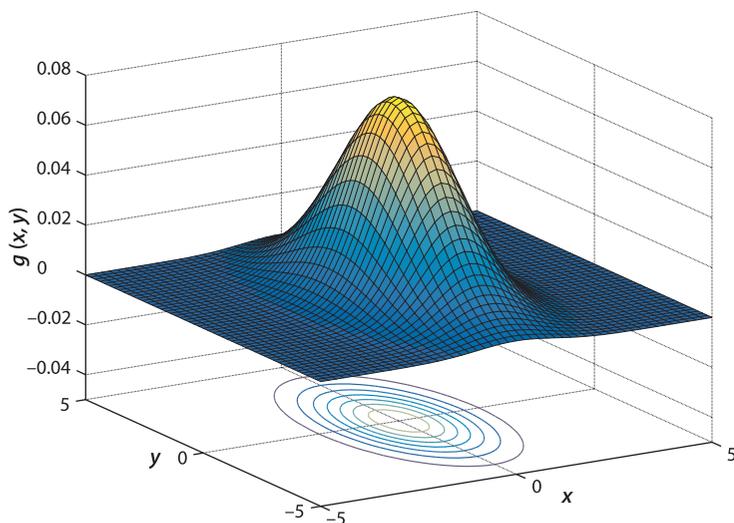


Fig. G.2. The 2-dimensional Gaussian with covariance $\mathbf{P} = \text{diag}(1^2, 2^2)$. Contour lines of constant probability density are shown beneath

It is also the definition of Mahalanobis distance, the covariance weighted distance between \mathbf{x} and $\boldsymbol{\mu}$.

The connection between Gaussian probability density functions and ellipses can be found in the quadratic exponent of Eq. G.2 which is the equation of an ellipse or ellipsoid[◀]. All the points that satisfy

$$(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{P}^{-1}(\mathbf{x} - \boldsymbol{\mu}) = s$$

result in a constant probability density value, that is, a contour of the 2-dimensional Gaussian. s is related to the probability by

$$s = \chi_n^2(p)$$

If we draw a vector of length n from the multivariate Gaussian each element is normally distributed. The sum of squares of independent normally distributed values is known to be distributed according to a χ^2 (chi-squared) distribution with n degrees of freedom.

which is the χ^2 distribution[◀] with n degrees of freedom, 2 in this case, and p is the probability that the point \mathbf{x} lies on the ellipse. For example the 50% confidence interval is

```
>> s = chi2inv(0.5, 2)
s =
    1.3863
```

where the first argument is the probability and the second is the number of degrees of freedom[▶].

This function requires the MATLAB Statistics and Machine Learning Toolbox[™]. The Robotics Toolbox provides `chi2inv_rtb` which is an approximation for the case $n=2$.

If the covariance matrix is diagonal then the ellipse is aligned with the x - and y -axes as we saw in Sect. C.1.4. This indicates that the two variables are independent and have zero correlation. Conversely a rotated ellipse indicates that the covariance is not diagonal and the two variables are correlated.

To draw a covariance ellipse we use the general approach for ellipses outlined in Sect. C.1.4 but the right-hand side of the ellipse equation is s not 1, and $\mathbf{E} \equiv \mathbf{P}$.

H

Kalman Filter

Consider the system shown in Fig. H.1. The physical robot is a “black box” which has a true state or pose x that evolves over time according to the applied inputs. We cannot directly measure the state, but sensors on the robot have outputs which are a function of that true state. Our challenge is: given the system inputs and sensor outputs *estimate* the unknown true state x and how certain we are of that estimate.

At face value this might seem hard, or even impossible, but there are quite a lot of things we know about system that will help us. Firstly, we know how the state evolves over time as a function of the inputs – this is the state transition model $f(\cdot)$, and we know the inputs to the system u . Our model is unlikely to be perfect and it is common to represent this uncertainty by an imaginary random number generator which is corrupting the system state – process noise. Secondly, we know how the sensor output depends on the state – this is the sensor model $h(\cdot)$ and its uncertainty is also modeled by an imaginary random number generator – sensor noise.

The imaginary random number sources v and w are *inside* the black box so the random numbers are also unknowable. However we can describe the characteristics of these random numbers – their *distribution* which tells us how likely it is that we will *draw* a random number with a particular value. A lot of noise in physical systems can be modeled well by the Gaussian (aka normal) distribution $\mathcal{N}(\mu, \sigma^2)$ which is characterized by a mean μ and a standard deviation σ . There are infinitely many possible distributions but the Gaussian distribution has some nice mathematical properties that we will rely on. However we should never assume that noise is Gaussian – we should attempt to determine the distribution by understanding the physics of the process and the sensor, or from careful measurement and analysis.

Often called the process or motion model.

For example wheel slippage on a mobile ground robot or wind gusts for a UAV.

Which can be nonsymmetrical or have multiple peaks.

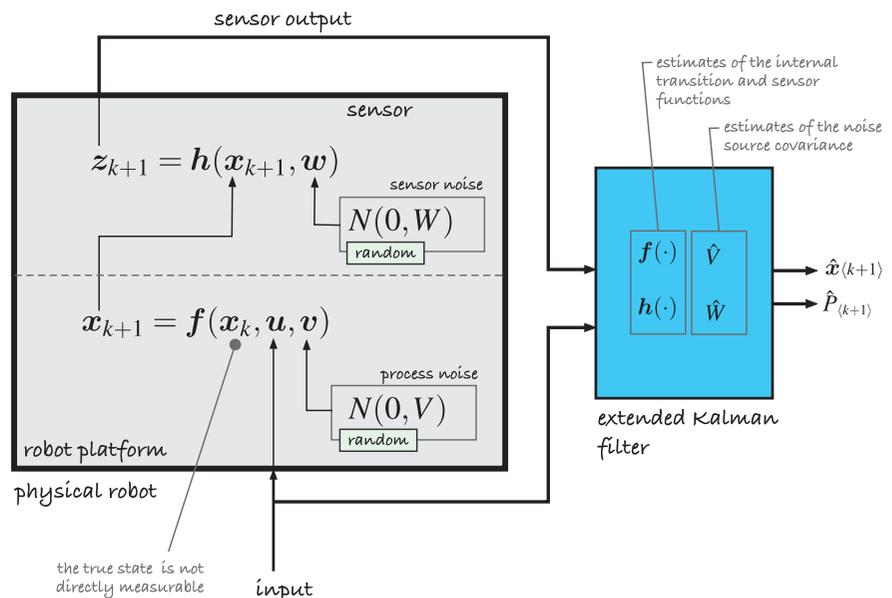


Fig. H.1.

The physical robot on the left has a true state that cannot be directly measured, however we gain a clue from the sensor output which is a function of this unknown true state

the true state is not directly measurable

input

In general terms, the problem we wish to solve is:

given a model of the system $f(\cdot)$, $h(\cdot)$, \hat{V} and \hat{W} ; the known inputs applied to the system u ; and some noisy sensor measurements z , find an estimate \hat{x} of the system state and our uncertainty \hat{P} in that estimate.

In a robotic localization context x is the unknown position or pose of the robot, u is the commands sent to the motors and z is the output of various sensors on the robot. For a ground robot x would be the pose in $SE(2)$ and u would be the motor commands and z might be the measured odometry or range and bearing to landmarks. For a flying robot x would be the pose in $SE(3)$ and u are the known forces applied to the airframe and z might be the measured accelerations and angular velocities. ▶

The state is a vector and there are many approaches to mapping pose to a vector, especially the rotational component – Euler angles, quaternions, and exponential coordinates are commonly used.

H.1 Linear Systems – Kalman Filter

Consider the transition model described as a discrete-time linear time-invariant system

$$x^{(k+1)} = Fx^{(k)} + Gu^{(k)} + v^{(k)} \quad (\text{H.1})$$

$$z^{(k)} = Hx^{(k)} + w^{(k)} \quad (\text{H.2})$$

where k is the time step, $x \in \mathbb{R}^n$ is the state vector, and $u \in \mathbb{R}^m$ is a vector of inputs to the system at time k , for example a velocity command, or applied forces and torques. The matrix $F \in \mathbb{R}^{n \times n}$ describes the dynamics of the system, that is, how the states evolve with time. The matrix $G \in \mathbb{R}^{n \times m}$ describes how the inputs are coupled to the system states. The vector $z \in \mathbb{R}^p$ represents the outputs of the system as measured by sensors. The matrix $H \in \mathbb{R}^{p \times n}$ describes how the system states are mapped to the system outputs which we can observe.

To account for errors in the motion model (F and G) or unmodeled disturbances we introduce a Gaussian random variable $v \in \mathbb{R}^n$ termed the process noise. $v^{(k)} \sim N(0, V)$, that is, it has zero mean and covariance $V \in \mathbb{R}^{n \times n}$. Covariance is a matrix quantity which is the variance for a multi-dimensional distribution – it is a positive definite matrix and therefore symmetric. The sensor measurement model H is not perfect either and this is modeled by sensor measurement noise, a Gaussian random variable $w \in \mathbb{R}^p$, $w^{(k)} \sim N(0, W)$ and covariance $W \in \mathbb{R}^{p \times p}$.

The Kalman filter is an optimal estimator for the case where the process and measurement noise are zero-mean Gaussian noise. The filter has two steps: prediction and update. The prediction is based on the previous state and the inputs that were applied

$$\hat{x}^{+(k+1)} = F\hat{x}^{(k)} + Gu^{(k)} \quad (\text{H.3})$$

$$\hat{P}^{+(k+1)} = \underbrace{F\hat{P}^{(k)}F^T}_{\text{projects}} + \hat{V} \quad (\text{H.4})$$

where \hat{x} is the estimate of the state and $\hat{P} \in \mathbb{R}^{n \times n}$ is the estimated covariance, or uncertainty, in \hat{x} . The notation $^+$ makes explicit that the left-hand side is an estimate at time $k + 1$ based on information from time k . \hat{V} is our best estimate of the covariance of the process noise.

The indicated term in Eq. H.4 *projects* the estimated covariance from the current time step to the next. Consider a one dimensional example where F is a scalar and the state estimate $\hat{x}^{(k)}$ has a PDF which is Gaussian with a mean $\bar{x}^{(k)}$ and a variance $\sigma^2(k)$. The prediction equation maps the state and its Gaussian distribution to a new Gaussian distribution with a mean $F\bar{x}^{(k)}$ and a variance $F^2\sigma^2(k)$. The term $F\hat{P}^{(k)}F^T$ is the matrix form of this since

$$\text{cov}(Fx) = F \text{cov}(x) F^T \quad (\text{H.5})$$

which scales the covariance appropriately.

The prediction of $\hat{\mathbf{P}}$ involves the addition of two positive-definite matrices so the uncertainty will increase – this is to be expected since we have used an uncertain model to predict the future value of an already uncertain estimate. $\hat{\mathbf{V}}$ must be a reasonable estimate of the covariance of the actual process noise. If we overestimate it, that is our estimate of process noise is larger than it really is, then we will have a large increase in uncertainty at this step, a pessimistic estimate of our certainty.

To counter this growth in uncertainty we need to introduce new information such as measurements made by the sensors since they depend on the state. The difference between what the sensors measure and what the sensors are predicted to measure is

$$\nu = \mathbf{z}^\# \langle k+1 \rangle - \mathbf{H} \hat{\mathbf{x}}^+ \langle k+1 \rangle \in \mathbb{R}^p$$

Some of this difference is due to noise in the sensor, the measurement noise, but the remainder provides valuable information related to the error between the actual and the predicted value of the state. Rather than considering this as error we refer to it more positively as *innovation* – new information.

The second step of the Kalman filter, the *update* step, maps the innovation into a correction for the predicted state, optimally tweaking the estimate based on what the sensors observed

$$\hat{\mathbf{x}} \langle k+1 \rangle = \hat{\mathbf{x}}^+ \langle k+1 \rangle + \mathbf{K} \nu \quad (\text{H.6})$$

$$\hat{\mathbf{P}} \langle k+1 \rangle = \hat{\mathbf{P}}^+ \langle k+1 \rangle - \mathbf{K} \mathbf{H} \hat{\mathbf{P}}^+ \langle k+1 \rangle \quad (\text{H.7})$$

Uncertainty is now *decreased* or *deflated*, since new information, from the sensors, is being incorporated. The matrix

$$\mathbf{K} = \mathbf{P}^+ \langle k+1 \rangle \mathbf{H}^T \left(\underbrace{\mathbf{H} \mathbf{P}^+ \langle k+1 \rangle \mathbf{H}^T + \hat{\mathbf{W}}}_{\text{innovation covariance}} \right)^{-1} \in \mathbb{R}^{n \times p} \quad (\text{H.8})$$

is known as the Kalman gain. The term indicated is the estimated covariance of the innovation and comprises the uncertainty in the state and the estimated measurement noise covariance. If the innovation has high uncertainty in some dimensions then the Kalman gain will be correspondingly small, that is, if the new information is uncertain then only small changes are made to the state vector. The term $\mathbf{H} \mathbf{P}^+ \langle k+1 \rangle \mathbf{H}^T$ in Eq. H.13 *projects* the covariance of the state estimate into the space of sensor values.

The covariance matrix must be positive-definite but after many updates the accumulated numerical errors may cause this matrix to be no longer symmetric. The positive-definite structure can be enforced by using the Joseph form of Eq. H.7

$$\hat{\mathbf{P}} \langle k+1 \rangle = (\mathbf{I}_{n \times n} - \mathbf{K} \mathbf{H}) \hat{\mathbf{P}}^+ \langle k+1 \rangle (\mathbf{I}_{n \times n} - \mathbf{K} \mathbf{H})^T + \mathbf{K} \hat{\mathbf{V}} \mathbf{K}^T$$

but this is computationally more costly.

The equations above constitute the classical Kalman filter which is widely used in robotics, aerospace and econometric applications. The filter has a number of important characteristics. Firstly it is optimal, but only if the noise is truly Gaussian with zero mean and time invariant parameters. This is often a good assumption but not always. Secondly it is recursive, the output of one iteration is the input to the next. Thirdly, it is asynchronous. At a particular iteration if no sensor information is available we just perform the prediction step and not the update. In the case that there are different sensors, each with their own \mathbf{H} , and different sample rates, we just apply the update with the appropriate \mathbf{z} and \mathbf{H} . The filter must be initialized with some reasonable value of $\hat{\mathbf{x}}$ and $\hat{\mathbf{P}}$, as well as good choices of the covariance matrices $\hat{\mathbf{V}}$ and $\hat{\mathbf{W}}$. As the filter runs the estimated covariance $\|\hat{\mathbf{P}}\|$ decreases but never reaches zero – the minimum value can be shown to be a function of $\hat{\mathbf{V}}$ and $\hat{\mathbf{W}}$. The Kalman-Bucy filter is a continuous-time version of this filter.

The covariance matrix \hat{P} is rich in information. The diagonal elements \hat{P}_{ii} are the variance, or uncertainty, in the state x_i . The off-diagonal elements \hat{P}_{ij} are the correlations between states x_i and x_j and indicate that the errors are not independent. The correlations are critical in allowing any piece of new information to *flow through* to adjust all the states that affect a particular process output.

H.2 Nonlinear Systems – Extended Kalman Filter

For the case where the system is not linear it can be described generally by two functions: the state transition (the motion model in robotics) and the sensor model

$$\mathbf{x}\langle k+1\rangle = \mathbf{f}(\mathbf{x}\langle k\rangle, \mathbf{u}\langle k\rangle, \mathbf{v}\langle k\rangle) \quad (\text{H.9})$$

$$\mathbf{z}\langle k\rangle = \mathbf{h}(\mathbf{x}\langle k\rangle, \mathbf{w}\langle k\rangle) \quad (\text{H.10})$$

and as before we represent model uncertainty, external disturbances and sensor noise by Gaussian random variables \mathbf{v} and \mathbf{w} .

We linearize the state transition function about the current state estimate $\hat{\mathbf{x}}_k$ as shown in Fig. H.2 resulting in

$$\mathbf{x}'\langle k+1\rangle \approx \mathbf{F}_x \mathbf{x}'\langle k\rangle + \mathbf{F}_u \mathbf{u}\langle k\rangle + \mathbf{F}_v \mathbf{v}\langle k\rangle \quad (\text{H.11})$$

$$\mathbf{z}'\langle k\rangle \approx \mathbf{H}_x \mathbf{x}'\langle k\rangle + \mathbf{H}_w \mathbf{w}\langle k\rangle \quad (\text{H.12})$$

where $\mathbf{F}_x = \partial \mathbf{f} / \partial \mathbf{x} \in \mathbb{R}^{n \times n}$, $\mathbf{F}_u = \partial \mathbf{f} / \partial \mathbf{u} \in \mathbb{R}^{n \times m}$, $\mathbf{F}_v = \partial \mathbf{f} / \partial \mathbf{v} \in \mathbb{R}^{n \times n}$, $\mathbf{H}_x = \partial \mathbf{h} / \partial \mathbf{x} \in \mathbb{R}^{p \times n}$ and $\mathbf{H}_w = \partial \mathbf{h} / \partial \mathbf{w} \in \mathbb{R}^{p \times p}$ are Jacobians of the functions $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$. Equating coefficients between Eq. H.1 and Eq. H.11 gives $\mathbf{F} \sim \mathbf{F}_x$, $\mathbf{G} \sim \mathbf{F}_u$ and $\mathbf{v}\langle k\rangle \sim \mathbf{F}_v \mathbf{v}\langle k\rangle$; and between Eq. H.2 and Eq. H.12 gives $\mathbf{H} \sim \mathbf{H}_x$ and $\mathbf{w}\langle k\rangle \sim \mathbf{H}_w \mathbf{w}\langle k\rangle$.

Taking the prediction equation Eq. H.9 with $\mathbf{v}\langle k\rangle = 0$, and the covariance equation Eq. H.4 with the linearized terms substituted we can write the prediction step as

$$\hat{\mathbf{x}}^+\langle k+1\rangle = \mathbf{f}(\hat{\mathbf{x}}\langle k\rangle, \mathbf{u}\langle k\rangle)$$

$$\hat{\mathbf{P}}^+\langle k+1\rangle = \mathbf{F}_x \hat{\mathbf{P}}\langle k\rangle \mathbf{F}_x^T + \mathbf{F}_v \hat{\mathbf{V}} \mathbf{F}_v^T$$

and the update step as

$$\hat{\mathbf{x}}\langle k+1\rangle = \hat{\mathbf{x}}^+\langle k+1\rangle + \mathbf{K}\nu$$

$$\hat{\mathbf{P}}\langle k+1\rangle = \hat{\mathbf{P}}^+\langle k+1\rangle - \mathbf{K} \mathbf{H}_x \hat{\mathbf{P}}^+\langle k+1\rangle$$

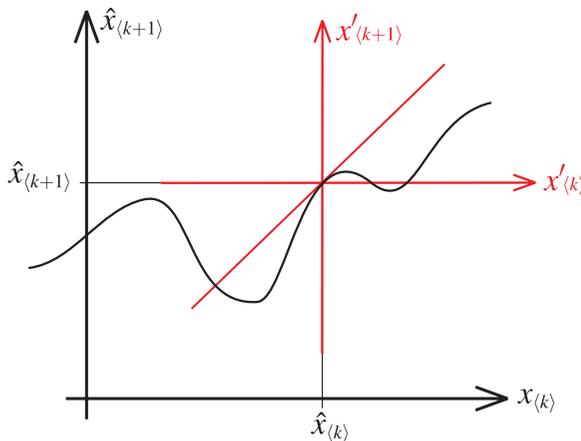


Fig. H.2. One dimensional example illustrating how the nonlinear state transition function $f: x_k \mapsto x_{k+1}$ shown in black is linearized about the point $(\hat{x}\langle k\rangle, \hat{x}\langle k+1\rangle)$ shown in red

Procedure EKF
Input : $\hat{\mathbf{x}}\langle k \rangle \in \mathbb{R}^n$, $\hat{\mathbf{P}}\langle k \rangle \in \mathbb{R}^{n \times n}$, $\mathbf{u}\langle k \rangle \in \mathbb{R}^m$, $\mathbf{z}\langle k+1 \rangle \in \mathbb{R}^p$, $\hat{\mathbf{V}} \in \mathbb{R}^{n \times n}$, $\hat{\mathbf{W}} \in \mathbb{R}^{p \times p}$
Output: $\hat{\mathbf{x}}\langle k+1 \rangle \in \mathbb{R}^n$, $\hat{\mathbf{P}}\langle k+1 \rangle \in \mathbb{R}^{n \times n}$

 – linearize about $\mathbf{x} = \hat{\mathbf{x}}\langle k \rangle$

 compute Jacobians: $\mathbf{F}_x \in \mathbb{R}^{n \times n}$, $\mathbf{F}_v \in \mathbb{R}^{n \times n}$, $\mathbf{H}_x \in \mathbb{R}^{p \times n}$, $\mathbf{H}_w \in \mathbb{R}^{p \times p}$

– the prediction step

$$\hat{\mathbf{x}}^+\langle k+1 \rangle = \mathbf{f}(\hat{\mathbf{x}}\langle k \rangle, \mathbf{u}\langle k \rangle) \quad // \text{predict state at next time step}$$

$$\hat{\mathbf{P}}^+\langle k+1 \rangle = \mathbf{F}_x \hat{\mathbf{P}}\langle k \rangle \mathbf{F}_x^T + \mathbf{F}_v \hat{\mathbf{V}} \mathbf{F}_v^T \quad // \text{predict covariance at next time step}$$

– the update step

$$\nu = \mathbf{z}\langle k+1 \rangle - \mathbf{h}(\hat{\mathbf{x}}^+\langle k+1 \rangle) \quad // \text{innovation : measured - predicted sensor value}$$

$$\mathbf{K} = \mathbf{P}^+\langle k+1 \rangle \mathbf{H}_x^T \left[\mathbf{H}_x \mathbf{P}^+\langle k+1 \rangle \mathbf{H}_x^T + \mathbf{H}_w \hat{\mathbf{W}} \mathbf{H}_w^T \right]^{-1} \quad // \text{Kalman gain}$$

$$\hat{\mathbf{x}}\langle k+1 \rangle = \hat{\mathbf{x}}^+\langle k+1 \rangle + \mathbf{K} \nu \quad // \text{update state estimate}$$

$$\hat{\mathbf{P}}\langle k+1 \rangle = \hat{\mathbf{P}}^+\langle k+1 \rangle - \mathbf{K} \mathbf{H}_x \hat{\mathbf{P}}^+\langle k+1 \rangle \quad // \text{update covariance estimate}$$

Algorithm H.1.
 Procedure EKF

where the Kalman gain is now

$$\mathbf{K} = \mathbf{P}^+\langle k+1 \rangle \mathbf{H}_x^T \left(\mathbf{H}_x \mathbf{P}^+\langle k+1 \rangle \mathbf{H}_x^T + \mathbf{H}_w \hat{\mathbf{W}} \mathbf{H}_w^T \right)^{-1} \quad (\text{H.13})$$

These equations are only valid at the linearization point $\hat{\mathbf{x}}\langle k \rangle$ – the Jacobians \mathbf{F}_x , \mathbf{F}_v , \mathbf{H}_x , \mathbf{H}_w must be computed at every iteration. ◀ The full procedure is summarized in Algorithm H.1.

A fundamental problem with the extended Kalman filter is that PDFs of the random variables are no longer Gaussian after being operated on by the nonlinear functions $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$. We can easily illustrate this by considering a nonlinear scalar function $y = (x + 2)^2 / 4$. We will draw a million Gaussian random numbers from the normal distribution $\mathcal{N}(5, 4)$ which has a mean of 5 and a standard deviation of 2

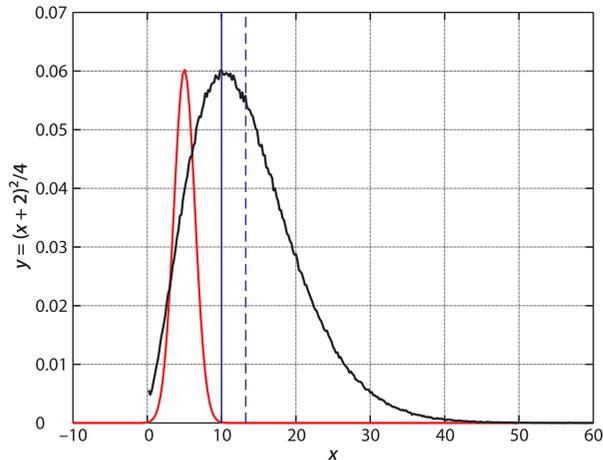
```
>> x = 2*randn(1000000,1) + 5;
```

and map them through our function

```
>> y = (x+2).^2 / 4;
```

and plot the probability density function of y

```
>> histogram(y, 'Normalization', 'pdf');
```


Fig. H.3.

PDF of the state x (red) which is Gaussian $\mathcal{N}(5, 4)$ and the PDF of the nonlinear function $y = (x + 2)^2 / 4$ (black). The peak and the mean of the non-linear distribution are shown by blue solid and dashed vertical lines respectively

Properly these matrices should be denoted as depending on the time step, i.e. $\mathbf{F}_x\langle k \rangle$ but this has been dropped in the interest of readability.

which is shown in Fig. H.3. We see that the PDF of y is substantially changed and no longer Gaussian. It has lost its symmetry so the mean value is greater than the mode. The Jacobians that appear in the EKF equations appropriately scale the covariance but the resulting non-Gaussian distribution breaks the assumptions which guarantee that the Kalman filter is an optimal estimator. Alternatives include the iterated EKF described by Jazwinski (2007) or the Unscented Kalman Filter (UKF) (Julier and Uhlmann 2004) or the sigma-point filter which uses discrete sample points (sigma points) to approximate the PDF.

Graphs

A graph is an abstract representation of a set of objects connected by links and depicted graphically as shown in Fig. I.1. Mathematically a graph is denoted $G(V, E)$ where V are the vertices or nodes, and E are the links that connect pairs of vertices and are called edges or arcs. Edges can be directed (*arrows*) or undirected as in this case. Edges can have an associated weight or cost associated with moving from one vertex to another. A sequence of edges from one vertex to another is a path, and a sequence that starts and ends at the same vertex is a cycle. An edge from a vertex to itself is a loop. Graphs can be used to represent transport, communications or social networks, and this branch of mathematics is graph theory.

The Toolbox provides a MATLAB® graph class called `PGraph` that supports embedded graphs where the vertices are associated with a point in an n -dimensional space. ◀ To create a new graph

This class is used other Toolbox classes such as `PRM`, `Lattice`, `RRT`, `PoseGraph` and `BundleAdjust`. MATLAB 2015b introduced a built in `graph` class to represent graphs.

```
>> g = PGraph()
g =
    2 dimensions
    0 vertices
    0 edges
    0 components
```

and by default the nodes of the graph exist in a 2-dimensional space. We can add nodes to the graph

```
>> g.add_node( rand(2,1) );
```

and each has a random coordinate. The `add_node` method returns an integer identifier for the node just added. A summary of the graph is given with its display method

```
>> g
g =
    2 dimensions
    5 vertices
    0 edges
    0 components
```

and shows that the graph has 5 nodes but no edges. The nodes are numbered 1 to 5 and we add edges between pairs of nodes

```
>> g.add_edge(1, 2);
>> g.add_edge(1, 3);
>> g.add_edge(1, 4);
>> g.add_edge(2, 3);
>> g.add_edge(2, 4);
>> g.add_edge(4, 5);
>> g
g =
    2 dimensions
    5 vertices
    6 edges
    1 components
```

By default the distance between the nodes is the Euclidean distance between the vertices but this can be overridden by a third argument to `add_edge`. The methods `add_node` and `add_edge` return an integer that uniquely identifies the node or edge just created. The graph has one component, that is all the nodes are connected into one network. The graph can be plotted by

```
>> g.plot('labels')
```

as shown in Fig. I.1. The vertices are shown as blue circles, and the option `'labels'` displays the vertex index next to the circle. Edges are shown as black lines joining vertices. Many options exist to change default plotting behavior. Note that only graphs embedded in 2- and 3-dimensional space can be plotted.

The neighbors of vertex 2 are

```
>> g.neighbours(2)
ans =
     3     4     1
```

which are vertices connected to vertex 2 by edges. Each edge has a unique index and the edges connecting to vertex 2 are

```
>> e = g.edges(2)
e =
     4     5     1
```

The cost or length of these edges is

```
>> g.cost(e)
ans =
 0.9597  0.3966  0.6878
```

and clearly edge 5 has a lower cost than edges 4 and 1. Edge 5

```
>> g.vertices(5)
ans =
     2     4
```

joins vertices 2 and 4, and vertex 4 is clearly the closest neighbor of vertex 2. Frequently we wish to obtain a node's neighboring vertices and their distances at the same time, and this can be achieved conveniently by

```
>> [n,c] = g.neighbours(2)
n =
     3     4     1
c =
 0.9597  0.3966  0.6878
```

Concise information about a node can be obtained by

```
>> g.about(1)
Node 1 #1@ (0.814724 0.905792 )
  neighbours:  >-o-> 2 3 4
    edges:    >-o-> 1 2 3
```

Arbitrary data can be attached to any node or edge by the methods `setvdata` and `setedata` respectively and retrieved by the methods `vdata` and `edata` respectively.

The vertex closest to the coordinate (0.5, 0.5) is

```
>> g.closest([0.5, 0.5])
ans =
     4
```

and the vertex closest to an interactively selected point is given by `g.pick`.

The minimum cost path between any two nodes in the graph can be computed using well known algorithms such as A* (Nilsson 1971)

```
>> g.Astar(3, 5)
ans =
     3     2     4     5
```

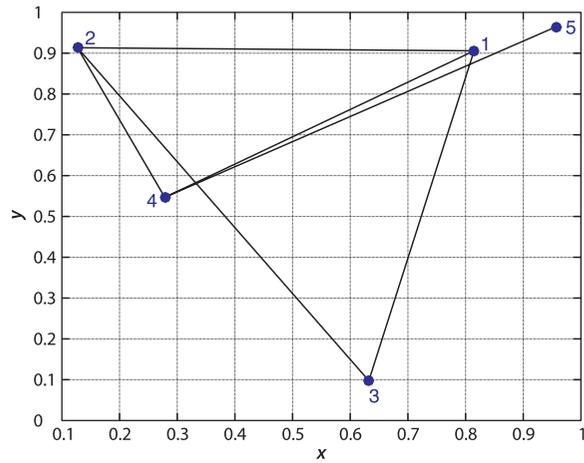


Fig. I.1.
An example graph generated by
the `PGraph` class

or the earlier method by Dijkstra (1959). By default the graph is treated as undirected, that is, the edges have no preferred direction. The `'directed'` option causes edges to be treated as directed, and the path will only traverse edges in their specified direction which is from the first to the second argument of the method `add_edge`.

Methods exist to compute various other representations of the graph such as adjacency, incidence, degree and Laplacian matrices.

J Peak Finding

A commonly encountered problem is estimating the position of the peak of some discrete 1-dimensional signal $y(k)$, $k \in \mathbb{Z}$, see for example Fig. J.1a

```
>> load peakfit1
>> plot(y, '-o')
```

Finding the peak to the nearest integer is straightforward using MATLAB's `max` function

```
>> [ypk,k] = max(y)
ypk =
    0.9905
k =
     8
```

which indicates the peak occurs at the eighth element and has a value of 0.9905. In this case there is more than one peak and we can use the Toolbox function `peak` instead

```
>> [ypk,k] = peak(y)
ypk =
    0.9905    0.6718   -0.5799
k =
     8     25     16
```

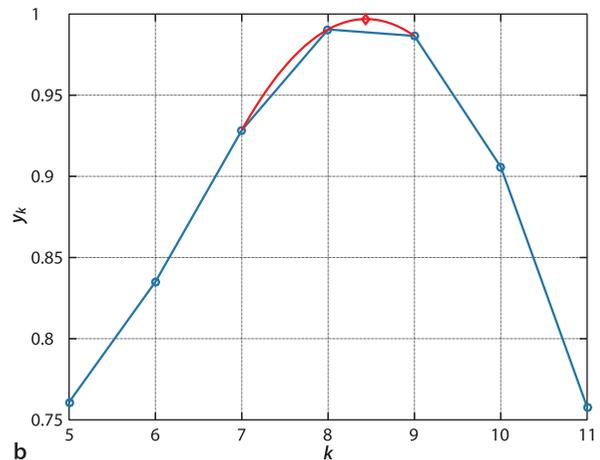
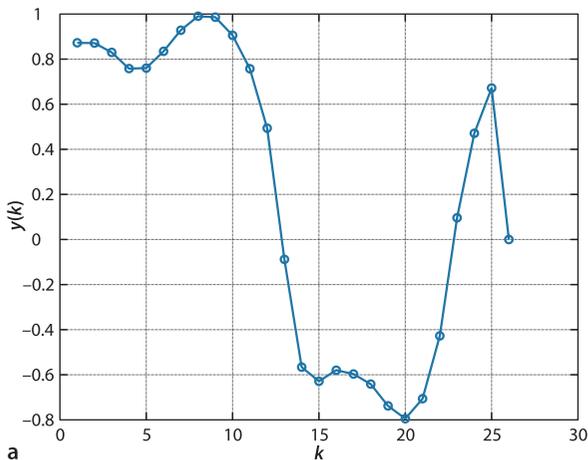
which has returned three maxima in descending magnitude. A common test of the quality of a peak is its magnitude and the ratio of the height of the second peak to the first peak

```
>> ypk(2)/ypk(1)
ans =
    0.6783
```

Fig. J.1. Peak fitting. **a** A signal with several local maxima; **b** closeup view of the first maxima with the fit curve (red) and the estimated peak (red- \diamond)

which is called the ambiguity ratio and is ideally small.

This signal is a sampled representation of a continuous underlying signal $y(x)$ and the real peak might actually lie between the samples. If we look at a zoomed version of the signal, Fig. J.1b, we can see that although the eighth point is the maximum the



ninth point is only slightly lower so the peak lies somewhere between points eight and nine. A common approach is to fit a parabola

$$y = a\delta_x^2 + b\delta_x + c, \quad \delta_x \in \mathbb{R} \quad (\text{J.1})$$

to the points surrounding the peak. For the discrete peak that occurs at (k, y_k) then $\delta_x = 0$ corresponds to k and the discrete x -coordinates on either side correspond to $\delta_x = -1$ and $\delta_x = +1$ respectively. Substituting the points $(k-1, y_{k-1})$, (k, y_k) and $(k+1, y_{k+1})$ into Eq. J.1 we can write three equations

$$\begin{aligned} y_{k-1} &= a - b + c \\ y_k &= c \\ y_{k+1} &= a + b + c \end{aligned}$$

or in compact matrix form as

$$\begin{pmatrix} y_{k-1} \\ y_k \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

and then solve for the parabolic coefficients

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}^{-1} \begin{pmatrix} y_{k-1} \\ y_k \\ y_{k+1} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & -2 & 1 \\ -1 & 0 & 1 \\ 0 & 2 & 0 \end{pmatrix} \begin{pmatrix} y_{k-1} \\ y_k \\ y_{k+1} \end{pmatrix} \quad (\text{J.2})$$

The maxima of the parabola occurs when its derivative is zero

$$2a\delta_x + b = 0$$

and substituting the values of a and b from Eq. J.2 we find the displacement of the peak of the fitted parabola with respect to the discrete maxima

$$\delta_x = \frac{1}{2} \frac{y_{k-1} - y_{k+1}}{y_{k-1} - 2y_k + y_{k+1}}, \quad \delta_x \in (-1, 1)$$

so the refined, or interpolated, position of the maxima is at

$$\hat{x} = k + \delta_x \in \mathbb{R}$$

and the estimated value of the maxima is obtained by substituting δ_x into Eq. J.1.

The coefficient a , which is negative for a maxima, indicates the sharpness of the peak which can be useful in determining whether a peak is *sufficiently* sharp. A large magnitude of a indicates a well defined sharp peak whereas a low value indicates a very broad peak for which estimation of a refined peak may not be so accurate.

Continuing the earlier example we can use the Toolbox function `peak` to estimate the refined peak positions

```
>> [ymax, xmax] = peak(y, 'interp', 2)
ymax =
    0.9905    0.6718   -0.5799
xmax =
    8.4394   24.7299   16.2438
```

where the argument after the `'interp'` option indicates that a second-order polynomial should be fitted. The fitted parabola is shown in red in Fig. J.1b and is plotted if the option `'plot'` is given.

If the signal has superimposed noise then there are likely to be multiple peaks, many of which are quite minor, and this can be overcome by specifying the *scale* of the peak. For example the peaks that are greater than all other values within ± 5 values in the horizontal direction are

```
>> peak(y, 'scale', 5)
ans =
    0.9905    0.8730    0.6718
```

In this case the result is unchanged since the signal is fairly smooth.

For a 2-dimensional signal we follow a similar procedure but instead fit a *paraboloid*

$$z = ax^2 + by^2 + cx + dy + e \quad (\text{J.3})$$

which has five coefficients that can be calculated from the center value (the discrete maximum) and its four neighbors (north, south, east and west) using a similar procedure to above. The displacement of the estimated peak with respect to the central point is

$$\delta_x = \frac{1}{2} \frac{z_e - z_w}{2z_c - z_w - z_e}, \quad \delta_x \in (-1, 1)$$

$$\delta_y = \frac{1}{2} \frac{z_s - z_n}{2z_c - z_n - z_s}, \quad \delta_y \in (-1, 1)$$

In this case the coefficients *a* and *b* represent the sharpness of the peak in the *x*- and *y*-directions, and the quality of the peak can be considered as being $\min(a, b)$.

A 2D discrete signal was loaded from `peakfit1` earlier

```
>> z
z =
   -0.0696    0.0348    0.1394    0.2436    0.3480
    0.0800    0.2000    0.3202    0.4400    0.5600
    0.0400    0.1717    0.3662    0.4117    0.5200
    0.0002    0.2062    0.8766    0.4462    0.4802
   -0.0400    0.0917    0.2862    0.3317    0.4400
   -0.0800    0.0400    0.1602    0.2800    0.4000
```

In this small example it is clear that the peak is at element (3, 4) using image coordinate convention, but programatically this is

```
>> [zmax, i] = max(z(:))
zmax =
    0.8766
i =
    16
```

and the maximum is at the sixteenth element in row-major order ◀ which we convert to array subscripts

```
>> [y, x] = ind2sub(size(z), i)
y =
     4
x =
     3
```

We can find this more conveniently using the Toolbox function `peak2`

```
>> [zpk, xy] = peak2(z)
zpk =
    0.8766    0.5600
xy =
     3     5
     4     2
```

Counting the elements, starting with 1 at the top-left down each column then back to the top of the next rightmost column.

which has returned two local maxima, one per column of the returned variables. This function will return all nonlocal maxima where the size of the local region is given by the 'scale' option. As for the 1-dimensional case we can refine the estimate of the peak

```
>> [zpk,xy]=peak2(z, 'interp')
Warning: Peak at (5,2) too close to edge of image
zpk =
    0.8839
xy =
    3.1090
    3.9637
```

that is, the peak is at element (3.1090, 3.9637). When this process is applied to image data it is referred to as subpixel interpolation.

Bibliography

- Achtelik MW (2014) Advanced closed loop visual navigation for micro aerial vehicles. Ph.D. thesis, ETH Zurich
- Agarwal S, Furukawa Y, Snively N, Simon I, Curless B, Seitz SM, Szeliski R (2011) Building Rome in a day. *Commun ACM* 54(10):105–112
- Agarwal P, Burgard W, Stachniss C (2014) Survey of geodetic mapping methods: Geodetic approaches to mapping and the relationship to graph-based SLAM. *IEEE Robot Autom Mag* 21(3):63–80
- Agrawal M, Konolige K, Blas M (2008) CenSurE: Center surround extremas for realtime feature detection and matching. In: Forsyth D, Torr P, Zisserman A (eds) *Lecture notes in computer science. Computer Vision – ECCV 2008*, vol 5 305. Springer-Verlag, Berlin Heidelberg, pp 102–115
- Albertos P, Mareels I (2010) *Feedback and control for everyone*. Springer-Verlag, Berlin Heidelberg
- Altmann SL (1989) Hamilton, Rodrigues, and the quaternion scandal. *Math Mag* 62(5):291–308
- Alton K, Mitchell IM (2006) Optimal path planning under defferent norms in continuous state spaces. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. pp 866–872
- Andersen N, Ravn O, Sørensen A (1993) Real-time vision based control of servomechanical systems. In: Chatila R, Hirzinger G (eds) *Lecture notes in control and information sciences. Experimental Robotics II*, vol 190. Springer-Verlag, Berlin Heidelberg, pp 388–402
- Andersson RL (1989) Dynamic sensing in a ping-pong playing robot. *IEEE T Robotic Autom* 5(6):728–739
- Antonelli G (2014) *Underwater robots: Motion and force control of vehicle-manipulator systems*, 3rd ed. Springer Tracts in Advanced Robotics, vol 2. Springer-Verlag, Berlin Heidelberg
- Arandjelovi R, Zisserman A (2012) Three things everyone should know to improve object retrieval. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp 2911–2918
- Arkin RC (1999) *Behavior-based robotics*. MIT Press, Cambridge, Massachusetts
- Armstrong WW (1979) Recursive solution to the equations of motion of an N-link manipulator. In: *Proceedings of the 5th World Congress on Theory of Machines and Mechanisms*, Montreal, Jul, pp 1 343–1 346
- Armstrong BS (1988) *Dynamics for robot control: Friction modelling and ensuring excitation during parameter identification*. Stanford University
- Armstrong B (1989) On finding exciting trajectories for identification experiments involving systems with nonlinear dynamics. *Int J Robot Res* 8(6):28
- Armstrong B, Khatib O, Burdick J (1986) The explicit dynamic model and inertial parameters of the Puma 560 Arm. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol 3. pp 510–518
- Armstrong-Hélouvy B, Dupont P, De Wit CC (1994) A survey of models, analysis tools and compensation methods for the control of machines with friction. *Automatica* 30(7):1 083–1 138
- Arun KS, Huang TS, Blostein SD (1987) Least-squares fitting of 2 3-D point sets. *IEEE T Pattern Anal* 9(5):699–700
- Asada H (1983) A geometrical representation of manipulator dynamics and its application to arm design. *J Dyn Syst-T ASME* 105:131
- Astolfi A (1999) Exponential stabilization of a wheeled mobile robot via discontinuous control. *J Dyn Syst-T ASME* 121(1):121–126
- Azarbayejani A, Pentland AP (1995) Recursive estimation of motion, structure, and focal length. *IEEE T Pattern Anal* 17(6):562–575
- Bailey T (n.d.) Software resources. University of Sydney. <http://www-personal.acfr.usyd.edu.au/tbailey>
- Bailey T, Durrant-Whyte H (2006) Simultaneous localization and mapping: Part II. *IEEE Robot Autom Mag* 13(3):108–117
- Bakthavatchalam M, Chaumette F, Tahri O (2015) An improved modelling scheme for photometric moments with inclusion of spatial weights for visual servoing with partial appearance/disappearance. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. pp 6037–6043
- Baldridge AM, Hook SJ, Grove CI, Rivera G (2009) The ASTER spectral library version 2.0. *Remote Sens Environ* 113(4):711–715
- Ball RS (1876) *The theory of screws: A study in the dynamics of a rigid body*. Hodges, Foster & Co., Dublin
- Ball RS (1908) *A treatise on spherical astronomy*. Cambridge University Press, New York

- Ballard DH (1981) Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recogn* 13(2):111–122
- Banks J, Corke PI (2001) Quantitative evaluation of matching methods and validity measures for stereo vision. *Int J Robot Res* 20(7):512–532
- Bar-Shalom Y, Fortmann T (1988) Tracking and data association. *Mathematics in science and engineering*, vol 182. Academic Press, London Oxford
- Bar-Shalom Y, Rong Li X, Thiagalingam Kirubarajan (2001) Estimation with applications to tracking and navigation. John Wiley & Sons, Inc., Chichester
- Bauer J, Sünderhauf N, Protzel P (2007) Comparing several implementations of two recently published feature detectors. In: *IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*. Toulouse
- Bay H, Ess A, Tuytelaars T, Van Gool L (2008) Speeded-up robust features (SURF). *Comput Vis Image Und* 110(3):346–359
- Benosman R, Kang SB (2001) Panoramic vision: Sensors, theory, and applications. Springer-Verlag, Berlin Heidelberg
- Benson KB (ed) (1986) *Television engineering handbook*. McGraw-Hill, New York
- Bertozzi M, Broggi A, Cardarelli E, Fedriga R, Mazzei L, Porta P (2011) VIAC expedition: Toward autonomous mobility. *IEEE Robot Autom Mag* 18(3):120–124
- Besl PJ, McKay HD (1992) A method for registration of 3-D shapes. *IEEE T Pattern Anal* 14(2): 239–256
- Bhat DN, Nayar SK (2002) Ordinal measures for image correspondence. *IEEE T Pattern Anal* 20(4): 415–423
- Biber P, Straßer W (2003) The normal distributions transform: A new approach to laser scan matching. In: *Proceedings of the IEEE/RSJ International Conference on intelligent robots and systems (IROS)*, vol 3. pp 2743–2748
- Bishop CM (2006) *Pattern recognition and machine learning*. Information science and statistics. Springer-Verlag, New York
- Blewitt M (2011) *Celestial navigation for yachtsmen*. Adlard Coles Nautical, London
- Bolles RC, Baker HH, Marimont DH (1987) Epipolar-plane image analysis: An approach to determining structure from motion. *Int J Comput Vision* 1(1):7–55, Mar
- Bolles RC, Baker HH, Hannah MJ (1993) The JISCT stereo evaluation. In: *Image Understanding Workshop: proceedings of a workshop held in Washington, DC apr 18–21, 1993*. Morgan Kaufmann, San Francisco, pp 263
- Bolton W (2015) *Mechatronics: Electronic control systems in mechanical and electrical engineering*, 6th ed. Pearson, Harlow
- Borenstein J, Everett HR, Feng L (1996) *Navigating mobile robots: Systems and techniques*. AK Peters, Ltd, Natick, MA, USA, Out of print and available at <http://www-personal.umich.edu/~johannb/Papers/pos96rep.pdf>
- Borgefors G (1986) Distance transformations in digital images. *Comput Vision Graph* 34(3):344–371
- Bostrom N (2016) *Superintelligence: Paths, dangers, strategies*. Oxford University Press, Oxford, 432 p
- Bouguet J-Y (2010) *Camera calibration toolbox for MATLAB®*. http://www.vision.caltech.edu/bouguetj/calib_doc
- Brady M, Hollerbach JM, Johnson TL, Lozano-Pérez T, Mason MT (eds) (1982) *Robot motion: Planning and control*. MIT Press, Cambridge, Massachusetts
- Braitenberg V (1986) *Vehicles: Experiments in synthetic psychology*. MIT Press, Cambridge, Massachusetts
- Bray H (2014) *You are here: From the compass to GPS, the history and future of how we find ourselves*. Basic Books, New York
- Brockett RW (1983) Asymptotic stability and feedback stabilization. In: Brockett RW, Millmann RS, Sussmann HJ (eds) *Progress in mathematics. Differential geometric control theory*, vol 27. pp 181–191
- Broida TJ, Chandrashekar S, Chellappa R (1990) Recursive 3-D motion estimation from a monocular image sequence. *IEEE T Aero Elec Sys* 26(4):639–656
- Brooks RA (1986) A robust layered control system for a mobile robot. *IEEE T Robotic Autom* 2(1):14–23
- Brooks RA (1989) A robot that walks: Emergent behaviors from a carefully evolved network. MIT AI Lab, Memo 1091
- Brown MZ, Burschka D, Hager GD (2003) Advances in computational stereo. *IEEE T Pattern Anal* 25(8):993–1008
- Brynjólfsson E, McAfee A (2014) *The second machine age: Work, progress, and prosperity in a time of brilliant technologies*. W. W. Norton & Co., New York
- Buehler M, Iagnemma K, Singh S (eds) (2007) *The 2005 DARPA grand challenge: The great robot race*. Springer Tracts in Advanced Robotics, vol 36. Springer-Verlag, Berlin Heidelberg
- Buehler M, Iagnemma K, Singh S (eds) (2010) *The DARPA urban challenge*. Tracts in Advanced Robotics, vol 56. Springer-Verlag, Berlin Heidelberg
- Bukowski R, Haynes LS, Geng Z, Coleman N, Santucci A, Lam K, Paz A, May R, DeVito M (1991) Robot hand-eye coordination rapid prototyping environment. In: *Proc ISIR*, pp 16.15–16.28

- Buttazzo GC, Allotta B, Fanizza FP (1993) Mousebuster: A robot system for catching fast moving objects by vision. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Atlanta, pp 932–937
- Calonder M, Lepetit V, Strecha C, Fua P (2010) BRIEF: Binary robust independent elementary features. In: Daniilidis K, Maragos P, Paragios N (eds) Lecture notes in computer science. Computer Vision – ECCV 2010, vol 6311. Springer-Verlag, Berlin Heidelberg, pp 778–792
- Canny JF (1983) Finding edges and lines in images. MIT, Artificial Intelligence Laboratory, AI-TR-720. Cambridge, MA
- Canny J (1987) A computational approach to edge detection. In: Fischler MA, Firschein O (eds) Readings in computer vision: Issues, problems, principles, and paradigms. Morgan Kaufmann, San Francisco, pp 184–203
- Censi A (2008) An ICP variant using a point-to-line metric. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp 19–25
- Chahl JS, Srinivasan MV (1997) Reflective surfaces for panoramic imaging. *Appl Optics* 31(36):8275–8285
- Chatfield K, Lempitsky VS, Vedaldi A, Zisserman A (2011) The devil is in the details: An evaluation of recent feature encoding methods. In: Proceedings of the British Machine Vision Conference 2011. 12 p
- Chaumette F (1990) La relation vision-commande: Théorie et application et des tâches robotiques. Ph.D. thesis, Université de Rennes 1
- Chaumette F (1998) Potential problems of stability and convergence in image-based and position-based visual servoing. In: Kriegman DJ, Hager GD, Morse AS (eds) Lecture notes in control and information sciences. The confluence of vision and control, vol 237. Springer-Verlag, Berlin Heidelberg, pp 66–78
- Chaumette F (2004) Image moments: A general and useful set of features for visual servoing. *IEEE T Robot Autom* 20(4):713–723
- Chaumette F, Hutchinson S (2006) Visual servo control 1: Basic approaches. *IEEE Robot Autom Mag* 13(4):82–90
- Chaumette F, Hutchinson S (2007) Visual servo control 2: Advanced approaches. *IEEE Robot Autom Mag* 14(1):109–118
- Chaumette F, Rives P, Espiau B (1991) Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Seoul, pp 2248–2253
- Chesi G, Hashimoto K (eds) (2010) Visual servoing via advanced numerical methods. Lecture notes in computer science, vol 401. Springer-Verlag, Berlin Heidelberg
- Chiaverini S, Sciavicco L, Siciliano B (1991) Control of robotic systems through singularities. Lecture notes in control and information sciences. Advanced Robot Control, Proceedings of the International Workshop on Nonlinear and Adaptive Control: Issues in Robotics, vol 162. Springer-Verlag, Berlin Heidelberg, pp 285–295
- Chiuso A, Favaro P, Jin H, Soatto S (2002) Structure from motion causally integrated over time. *IEEE T Pattern Anal* 24(4):523–535
- Choset HM, Lynch KM, Hutchinson S, Kantor G, Burgard W, Kavraki LE, Thrun S (2005) Principles of robot motion. MIT Press, Cambridge, Massachusetts
- Colicchia G, Waltner C, Hopf M, Wiesner H (2009) The scallop's eye – A concave mirror in the context of biology. *Physics Education* 44(2):175–179
- Collewet C, Marchand E, Chaumette F (2008) Visual servoing set free from image processing. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA). pp 81–86
- Commission Internationale de L'Éclairage (1987) Colorimetry, 2nd ed. Commission Internationale de L'Éclairage, CIE No 15.2
- Corke PI (1994) High-performance visual closed-loop robot control. University of Melbourne, Dept. Mechanical and Manufacturing Engineering. <http://eprints.unimelb.edu.au/archive/00000547/01/thesis.pdf>
- Corke PI (1996a) In situ measurement of robot motor electrical constants. *Robotica* 14(4):433–436
- Corke PI (1996b) Visual control of robots: High-performance visual servoing. *Mechatronics*, vol 2. Research Studies Press (John Wiley). Out of print and available at <http://www.petercorke.com/bluebook>
- Corke PI (2001) Mobile robot navigation as a planar visual servoing problem. In: Jarvis RA, Zelinsky A (eds) Springer tracts in advanced robotics. *Robotics Research: The 10th International Symposium*, vol 6. IFRR, Lorne, pp 361–372
- Corke PI (2007) A simple and systematic approach to assigning Denavit-Hartenberg parameters. *IEEE T Robot Autom* 23(3):590–594
- Corke PI (2010) Spherical image-based visual servo and structure estimation. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Anchorage, pp 5550–5555
- Corke PI, Armstrong-Hélouvry BS (1994) A search for consensus among model parameters reported for the PUMA 560 robot. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). San Diego, pp 1608–1613
- Corke PI, Armstrong-Hélouvry BS (1995) A meta-study of PUMA 560 dynamics: A critical appraisal of literature data. *Robotica* 13(3):253–258

- Corke PI, Good MC (1992) Dynamic effects in high-performance visual servoing. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Nice, pp 1 838–1 843
- Corke PI, Good MC (1996) Dynamic effects in visual closed-loop systems. *IEEE T Robotic Autom* 12(5):671–683
- Corke PI, Hutchinson SA (2001) A new partitioned approach to image-based visual servo control. *IEEE T Robotic Autom* 17(4):507–515
- Corke PI, Dunn PA, Banks JE (1999) Frame-rate stereopsis using non-parametric transforms and programmable logic. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Detroit, pp 1928–1933
- Corke PI, Strelow D, Singh S (2004) Omnidirectional visual odometry for a planetary rover. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). Sendai, pp 4 007–4 012
- Corke PI, Spindler F, Chaumette F (2009) Combining Cartesian and polar coordinates in IBVS. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). St. Louis, pp 5 962–5 967
- Corke PI, Paul R, Churchill W, Newman P (2013) Dealing with shadows: Capturing intrinsic scene appearance for image-based outdoor localisation. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp 2085–2092
- Craig JJ (1987) Adaptive control of mechanical manipulators. Addison-Wesley Longman Publishing Co., Inc. Boston
- Craig JJ (2005) Introduction to robotics: Mechanics and control, 3rd ed. Pearson/Prentice Hall, Upper Saddle River, New Jersey
- Craig JJ, Hsu P, Sastry SS (1987) Adaptive control of mechanical manipulators. *Int J Robot Res* 6(2):16–28
- Crombez N, Caron G, Mouaddib EM (2015) Photometric Gaussian mixtures based visual servoing. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp 5486–5491
- Crone RA (1999) A history of color: The evolution of theories of light and color. Kluwer Academic, Dordrecht
- Cummins M, Newman P (2008) FAB-MAP: Probabilistic localization and mapping in the space of appearance. *Int J Robot Res* 27(6):647
- Cutting JE (1997) How the eye measures reality and virtual reality. *Behav Res Meth Ins C* 29(1):27–36
- Daniilidis K, Klette R (eds) (2006) Imaging beyond the pinhole camera. Computational Imaging, vol 33. Springer-Verlag, Berlin Heidelberg
- Dansereau DG (2014) Plenoptic signal processing for robust vision in field robotics. Ph.D. thesis, The University of Sydney
- Davison AJ, Reid ID, Molton ND, Stasse O (2007) MonoSLAM: Real-time single camera SLAM. *IEEE T Pattern Anal* 29(6):1 052–1 067
- Deguchi K (1998) Optimal motion control for image-based visual servoing by decoupling translation and rotation. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). Victoria, Canada, pp 705–711
- Dellaert F, Kaess M (2006) Square root SAM: Simultaneous localization and mapping via square root information smoothing. *Int J Robot Res* 25(12):1181–1203
- Dellaert F, Seitz SM, Thorpe CE, Thrun S (2000) Structure from motion without correspondence. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Hilton Head Island, SC, pp 557–564
- DeMenthon D, Davis LS (1992) Exact and approximate solutions of the perspective-three-point problem. *IEEE T Pattern Anal* 14(11):1 100–1 105
- Denavit J, Hartenberg RS (1955) A kinematic notation for lower-pair mechanisms based on matrices. *J Appl Mech-T ASME* 22(1):215–221
- Deo AS, Walker ID (1995) Overview of damped least-squares methods for inverse kinematics of robot manipulators. *J Intell Robot Syst* 14(1):43–68
- Deriche R, Giraudon G (1993) A computational approach for corner and vertex detection. *Int J Comput Vision* 10(2):101–124
- DeWitt BA, Wolf PR (2000) Elements of photogrammetry (with applications in GIS). McGraw-Hill, New York
- Dickmanns ED (2007) Dynamic vision for perception and control of motion. Springer-Verlag, London
- Dickmanns ED, Graefe V (1988a) Applications of dynamic monocular machine vision. *Mach Vision Appl* 1:241–261
- Dickmanns ED, Graefe V (1988b) Dynamic monocular machine vision. *Mach Vision Appl* 1(4):223–240
- Dickmanns ED, Zapp A (1987) Autonomous high speed road vehicle guidance by computer vision. In: Tenth Triennial World Congress of the International Federation of Automatic Control, vol 4. Munich, pp 221–226
- Dijkstra EW (1959) A note on two problems in connexion with graphs. *Numer Math* 1(1):269–271
- Dougherty ER, Lotufo RA (2003) Hands-on morphological image processing. Society of Photo-Optical Instrumentation Engineers (SPIE)

- Duda RO, Hart PE (1972) Use of the Hough transformation to detect lines and curves in pictures. *Commun ACM* 15(1):11–15
- Durrant-Whyte H, Bailey T (2006) Simultaneous localization and mapping: Part I. *IEEE Robot Autom Mag* 13(2):99–110
- Espiou B, Chaumette F, Rives P (1992) A new approach to visual servoing in robotics. *IEEE T Robotic Autom* 8(3):313–326
- Everett HR (1995) *Sensors for mobile robots: Theory and application*. AK Peters Ltd., Wellesley
- Faugeras OD (1993) *Three-dimensional computer vision: A geometric viewpoint*. MIT Press, Cambridge, Massachusetts
- Faugeras OD, Lustman F (1988) Motion and structure from motion in a piecewise planar environment. *Int J Pattern Recogn* 2(3):485–508
- Faugeras O, Luong QT, Papadopoulou T (2001) *The geometry of multiple images: The laws that govern the formation of images of a scene and some of their applications*. MIT Press, Cambridge, Massachusetts
- Featherstone R (1987) *Robot dynamics algorithms*. Kluwer Academic, Dordrecht
- Feddema JT (1989) Real time visual feedback control for hand-eye coordinated robotic systems. *Purdue University*
- Feddema JT, Mitchell OR (1989) Vision-guided servoing with feature-based trajectory generation. *IEEE T Robotic Autom* 5(5):691–700
- Feddema JT, Lee CSG, Mitchell OR (1991) Weighted selection of image features for resolved rate visual feedback control. *IEEE T Robotic Autom* 7(1):31–47
- Felzenszwalb PF, Huttenlocher DP (2004) Efficient graph-based image segmentation. *Int J Comput Vision* 59(2):167–181
- Ferguson D, Stentz A (2006) Using interpolation to improve path planning: The Field D* algorithm. *J Field Robotics* 23(2):79–101
- Fischler MA, Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 24(6):381–395
- Flusser J (2000) On the independence of rotation moment invariants. *Pattern Recogn* 33(9):1405–1410
- Fomena R, Chaumette F (2007) Visual servoing from spheres using a spherical projection model. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Rome, pp 2080–2085
- Ford M (2015) *Rise of the robots: Technology and the threat of a jobless future*. Basic Books, New York
- Förstner W (1994) A framework for low level feature extraction. In: Ecklundh J-O (ed) *Lecture notes in computer science. Computer Vision – ECCV 1994, vol 800*. Springer-Verlag, Berlin Heidelberg, pp 383–394
- Förstner W, Gülch E (1987) A fast operator for detection and precise location of distinct points, corners and centres of circular features. In: *ISPRS Intercommission Workshop*. Interlaken, pp 149–155
- Forsyth DA, Ponce J (2011) *Computer vision: A modern approach*, 2nd ed. Pearson, London
- Fraundorfer F, Scaramuzza D (2012) Visual odometry: Part II – Matching, robustness, optimization, and applications. *IEEE Robot Autom Mag* 19(2):78–90
- Freeman H (1974) Computer processing of line-drawing images. *ACM Comput Surv* 6(1):57–97
- Friedman DP, Felleisen M, Bibby D (1987) *The little LISPer*. MIT Press, Cambridge, Massachusetts
- Funda J, Taylor RH, Paul RP (1990) On homogeneous transforms, quaternions, and computational efficiency. *IEEE T Robotic Autom* 6(3):382–388
- Gans NR, Hutchinson SA, Corke PI (2003) Performance tests for visual servo control systems, with application to partitioned approaches to visual servo control. *Int J Robot Res* 22(10–11):955
- Gautier M, Khalil W (1992) Exciting trajectories for the identification of base inertial parameters of robots. *Int J Robot Res* 11(4):362
- Geiger A, Roser M, Urtasun R (2010) Efficient large-scale stereo matching. In: Kimmel R, Klette R, Sugimoto A (eds) *Computer vision – ACCV 2010: 10th Asian Conference on Computer Vision, Queenstown, New Zealand, November 8–12, 2010, revised selected papers, part I*. Springer-Verlag, Berlin Heidelberg, pp 25–38
- Geraerts R, Overmars MH (2004) A comparative study of probabilistic roadmap planners. In: Boissonnat J-D, Burdick J, Goldberg K, Hutchinson S (eds) *Springer tracts in advanced robotics. Algorithmic Foundations of Robotics V, vol 7*. Springer-Verlag, Berlin Heidelberg, pp 43–58
- Gevers T, Gijzenij A, van de Weijer J, Geusebroek J-M (2012) *Color in computer vision: Fundamentals and applications*. John Wiley & Sons, Inc., Chichester
- Geyer C, Daniilidis K (2000) A unifying theory for central panoramic systems and practical implications. In: Vernon D (ed) *Lecture notes in computer science. Computer vision – ECCV 2000, vol 1843*. Springer-Verlag, Berlin Heidelberg, pp 445–461
- Glover A, Maddern W, Warren M, Reid S, Milford M, Wyeth G (2012) OpenFABMAP: An open source toolbox for appearance-based loop closure detection. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. pp 4730–4735
- Gonzalez R, Woods R (2008) *Digital image processing*, 3rd ed. Prentice Hall, Upper Saddle River, New Jersey

- Gonzalez R, Woods R, Eddins S (2009) Digital image processing using MATLAB, 2nd ed. Gatesmark Publishing
- Grassia FS (1998) Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools* 3(3):29–48
- Gregory RL (1997) *Eye and brain: The psychology of seeing*. Princeton University Press, Princeton, New Jersey
- Grey CGP (2014) Humans need not apply. YouTube video, www.youtube.com/watch?v=7Pq-S557XQU
- Grisetti G (n.d.) Teaching resources. Sapienza University of Rome. <http://www.dis.uniroma1.it/~grisetti/teaching.html>
- Groves PD (2013) *Principles of GNSS, inertial, and multisensor integrated navigation systems*, 2nd ed. Artech House, Norwood, USA
- Hager GD, Toyama K (1998) X Vision: A portable substrate for real-time vision applications. *Comput Vis Image Und* 69(1):23–37
- Hamel T, Mahony R (2002) Visual servoing of an under-actuated dynamic rigid-body system: An image based approach. *IEEE T Robot Autom* 18(2):187–198
- Hamel T, Mahony R, Lozano R, Ostrowski J (2002) Dynamic modelling and configuration stabilization for an X4-flyer. *IFAC World Congress* 1(2), p 3
- Hansen P, Corke PI, Boles W (2010) Wide-angle visual feature matching for outdoor localization. *Int J Robot Res* 29(1–2):267–297
- Harris CG, Stephens MJ (1988) A combined corner and edge detector. In: *Proceedings of the Fourth Alvey Vision Conference*. Manchester, pp 147–151
- Hart PE (2009) How the Hough transform was invented [DSP history]. *IEEE Signal Proc Mag* 26(6):18–22
- Hartenberg RS, Denavit J (1964) Kinematic synthesis of linkages. McGraw-Hill, New York, available online at <http://kmoddl.library.cornell.edu/bib.php?m=23>
- Hartley R, Zisserman A (2003) *Multiple view geometry in computer vision*. Cambridge University Press, New York
- Harvey P (nd) ExifTool. <http://www.sno.phy.queensu.ca/~phil/exiftool>
- Hashimoto K (ed) (1993) Visual servoing. In: *Robotics and automated systems*, vol 7. World Scientific, Singapore
- Hashimoto K, Kimoto T, Ebine T, Kimura H (1991) Manipulator control with image-based visual servo. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Seoul, pp 2267–2272
- Hellerstein JL, Diao Y, Parekh S, Tilbury DM (2004) *Feedback control of computing systems*. Wiley-IEEE Press, 456 p
- Herschel W (1800) Experiments on the refrangibility of the invisible rays of the sun. *Phil Trans R Soc Lond* 90:284–292
- Hill J, Park WT (1979) Real time control of a robot with a mobile camera. In: *Proceedings of the 9th ISIR*, SME. Washington, DC. Mar, pp 233–246
- Hirata T (1996) A unified linear-time algorithm for computing distance maps. *Inform Process Lett* 58(3):129–133
- Hirschmüller H (2008) Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(2):328–341
- Hirt C, Claessens S, Fecher T, Kuhn M, Pail R, Rexer M (2013) New ultrahigh-resolution picture of Earth's gravity field. *Geophys Res Lett* 40:4279–4283
- Hoag D (1963) Consideration of Apollo IMU gimbal lock. MIT Instrumentation Laboratory, E-1344, <http://www.hq.nasa.gov/alsj/e-1344.htm>
- Hollerbach JM (1980) A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. *IEEE T Syst Man Cyb* 10(11):730–736, Nov
- Hollerbach JM (1982) Dynamics. In: Brady M, Hollerbach JM, Johnson TL, Lozano-Pérez T, Mason MT (eds) *Robot motion – Planning and control*. MIT Press, Cambridge, Massachusetts, pp 51–71
- Horau R, Canio B, Leboulloux O (1989) An analytic solution for the perspective 4-point problem. *Comput Vision Graph* 47(1):33–44
- Horn BKP (1987) Closed-form solution of absolute orientation using unit quaternions. *J Opt Soc Am A* 4(4):629–642
- Horn BKP, Hilden HM, Negahdaripour S (1988) Closed-form solution of absolute orientation using orthonormal matrices. *J Opt Soc Am A* 5(7):1127–1135
- Hosoda K, Asada M (1994) Versatile visual servoing without knowledge of true Jacobian. In: *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*. Munich, pp 186–193
- Howard TM, Green CJ, Kelly A, Ferguson D (2008) State space sampling of feasible motions for high-performance mobile robot navigation in complex environments. *J Field Robotics* 25(6–7):325–345
- Hu MK (1962) Visual pattern recognition by moment invariants. *IRE T Inform Theor* 8:179–187
- Hua M-D, Ducard G, Hamel T, Mahony R, Rudin K (2014) Implementation of a nonlinear attitude estimator for aerial robotic vehicles. *IEEE T Contr Syst T* 22(1):201–213
- Huang TS, Netravali AN (1994) Motion and structure from feature correspondences: A review. *P IEEE* 82(2):252–268

- Humenberger M, Zinner C, Kubinger W (2009) Performance evaluation of a census-based stereo matching algorithm on embedded and multi-core hardware. In: Proceedings of the 19th International Symposium on Image and Signal Processing and Analysis (ISPA). pp 388–393
- Hunt RWG (1987) The reproduction of colour, 4th ed. Fountain Press, Tolworth
- Hunter RS, Harold RW (1987) The measurement of appearance. John Wiley & Sons, Inc., Chichester
- Hutchinson S, Hager G, Corke PI (1996) A tutorial on visual servo control. *IEEE T Robotic Autom* 12(5):651–670
- Iwatsuki M, Okiyama N (2002a) A new formulation of visual servoing based on cylindrical coordinate system with shiftable origin. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). Lausanne, pp 354–359
- Iwatsuki M, Okiyama N (2002b) Rotation-oriented visual servoing based on cylindrical coordinates. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Washington, DC, May, pp 4 198–4203
- Izaguirre A, Paul RP (1985) Computation of the inertial and gravitational coefficients of the dynamics equations for a robot manipulator with a load. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Mar, pp 1 024–1 032
- Jägersand M, Fuentes O, Nelson R (1996) Experimental evaluation of uncalibrated visual servoing for precision manipulation. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Albuquerque, NM, pp 2 874–2 880
- Jarvis RA, Byrne JC (1988) An automated guided vehicle with map building and path finding capabilities. In: *Robotics Research: The Fourth international symposium*. MIT Press, Cambridge, Massachusetts, pp 497–504
- Jazwinski AH (2007) *Stochastic processes and filtering theory*. Dover Publications, Mineola
- Jebara T, Azarbayejani A, Pentland A (1999) 3D structure from 2D motion. *IEEE Signal Proc Mag* 16(3):66–84
- Julier SJ, Uhlmann JK (2004) Unscented filtering and nonlinear estimation. *P IEEE* 92(3):401–422
- Kaehler A, Bradski G (2016) *Learning OpenCV: Computer vision in C++ with the OpenCV library*. O'Reilly & Associates, Köln
- Kaess M, Ranganathan A, Dellaert F (2007) iSAM: Fast incremental smoothing and mapping with efficient data association. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp 1670–1677
- Kahn ME (1969) The near-minimum time control of open-loop articulated kinematic linkages. Stanford University, AIM-106
- Kálmán RE (1960) A new approach to linear filtering and prediction problems. *J Basic Eng-T Asme* 82(1):35–45
- Kane TR, Levinson DA (1983) The use of Kane's dynamical equations in robotics. *Int J Robot Res* 2(3):3–21
- Karaman S, Walter MR, Perez A, Frazzoli E, Teller S (2011) Anytime motion planning using the RRT*. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp 1478–1483
- Kavraki LE, Svestka P, Latombe JC, Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE T Robotic Autom* 12(4):566–580
- Kelly R (1996) Robust asymptotically stable visual servoing of planar robots. *IEEE T Robotic Autom* 12(5):759–766
- Kelly A (2013) *Mobile robotics: Mathematics, models, and methods*. Cambridge University Press, New York
- Kelly R, Carelli R, Nasisi O, Kuchen B, Reyes F (2002a) Stable visual servoing of camera-in-hand robotic systems. *IEEE-ASME T Mech* 5(1):39–48
- Kelly R, Shirkey P, Spong MW (2002b) Fixed-camera visual servo control for planar robots. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Washington, DC, pp 2 643–2 649
- Khalil W, Creusot D (1997) SYMORO+: A system for the symbolic modelling of robots. *Robotica* 15(2):153–161
- Khalil W, Dombre E (2002) *Modeling, identification and control of robots*. Kogan Page Science, London
- Khatib O (1987) A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE T Robotic Autom* 3(1):43–53
- King-Hele D (2002) Erasmus Darwin's improved design for steering carriages and cars. *Notes and Records of the Royal Society of London* 56(1):41–62
- Klafter RD, Chmielewski TA, Negin M (1989) *Robotic engineering – An integrated approach*. Prentice Hall, Upper Saddle River, New Jersey
- Klein CA, Huang CH (1983) Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE T Syst Man Cyb* 13:245–250
- Klein G, Murray D (2007) Parallel tracking and mapping for small AR workspaces. In: Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2007). pp 225–234
- Klette R, Kruger N, Vaudrey T, Pauwels K, van Hulle M, Morales S, Kandil F, Haeusler R, Pugeault N, Rabe C (2011) Performance of correspondence algorithms in vision-based driver assistance using an online image sequence database. *IEEE T Veh Technol* 60(5):2 012–2 026

- Koenderink JJ (1984) The structure of images. *Biol Cybern* 50(5):363–370
- Koenderink JJ (2010) *Color for the sciences*. MIT Press, Cambridge, Massachusetts
- Koenig S, Likhachev M (2002) D^{*} Lite. In: *Proceedings of the National Conference on Artificial Intelligence*, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press, Cambridge, Massachusetts; 1999, pp 476–483
- Koenig S, Likhachev M (2005) Fast replanning for navigation in unknown terrain. *IEEE T Robot Autom* 21(3):354–363
- Kriegman DJ, Hager GD, Morse AS (eds) (1998) *The confluence of vision and control. Lecture notes in control and information sciences*, vol 237. Springer-Verlag, Berlin Heidelberg
- Kuipers JB (1999) *Quaternions and rotation sequences: A primer with applications to orbits, aerospace and virtual reality*. Princeton University Press, Princeton, New Jersey
- Kümmerle R, Grisetti G, Strasdat H, Konolige K, Burgard W (2011) g²o: A general framework for graph optimization. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. pp 3607–3613
- Lam O, Dayoub F, Schulz R, Corke P (2015) Automated topometric graph generation from floor plan analysis. In: *Proceedings of the Australasian Conference on Robotics and Automation. Australasian Robotics and Automation Association (ARAA)*
- Lamport L (1994) *LATEX: A document preparation system. User's guide and reference manual*. Addison-Wesley Publishing Company, Reading
- Land EH, McCann J (1971) Lightness and retinex theory. *J Opt Soc Am A* 61(1):1–11
- Land MF, Nilsson D-E (2002) *Animal eyes*. Oxford University Press, Oxford
- LaValle SM (1998) Rapidly-exploring random trees: A new tool for path planning. *Computer Science Dept., Iowa State University*, TR 98–11
- LaValle SM (2006) *Planning algorithms*. Cambridge University Press, New York
- LaValle SM (2011a) Motion planning: The essentials. *IEEE Robot Autom Mag* 18(1):79–89
- LaValle SM (2011b) Motion planning: Wild frontiers. *IEEE Robot Autom Mag* 18(2):108–118
- LaValle SM, Kuffner JJ (2001) Randomized kinodynamic planning. *Int J Robot Res* 20(5):378–400
- Laussedat A (1899) *La métrophotographie. Enseignement supérieur de la photographie*. Gauthier-Villars, 52 p
- Leavers VF (1993) Which Hough transform? *Comput Vis Image Und* 58(2):250–264
- Lee CSG, Lee BH, Nigham R (1983) Development of the generalized D'Alembert equations of motion for mechanical manipulators. In: *Proceedings of the 22nd CDC*, San Antonio, Texas. pp 1205–1210
- Lepetit V, Moreno-Noguer F, Fua P (2009) EPnP: An accurate O(*n*) solution to the PnP problem. *Int J Comput Vision* 81(2):155–166
- Li H, Hartley R (2006) Five-point motion estimation made easy. In: *18th International Conference on Pattern Recognition ICPR 2006*. Hong Kong, pp 630–633
- Li Y, Jia W, Shen C, van den Hengel A (2014) Characterness: An indicator of text in the wild. *IEEE T Image Process* 23(4):1666–1677
- Li T, Bolic M, Djuric P (2015) Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Proc Mag* 32(3):70–86
- Lin Z, Zeman V, Patel RV (1989) On-line robot trajectory planning for catching a moving object. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. pp 1726–1731
- Lindeberg T (1993) *Scale-space theory in computer vision*. Springer-Verlag, Berlin Heidelberg
- Lloyd J, Hayward V (1991) Real-time trajectory generation using blend functions. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Seoul, pp 784–789
- Longuet-Higgins H (1981) A computer algorithm for reconstruction of a scene from two projections. *Nature* 293:133–135
- Lovell J, Kluger J (1994) *Apollo 13*. Coronet Books
- Lowe DG (1991) Fitting parametrized three-dimensional models to images. *IEEE T Pattern Anal* 13(5): 441–450
- Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vision* 60(2):91–110
- Lowry S, Sunderhauf N, Newman P, Leonard J, Cox D, Corke P, Milford M (2015) Visual place recognition: A survey. *Robotics, IEEE Transactions on* (99):1–19
- Lu F, Milios E (1997) Globally consistent range scan alignment for environment mapping. *Auton Robot* 4:333–349
- Lucas SM (2005) ICDAR 2005 text locating competition results. In: *Proceedings of the Eighth International Conference on Document Analysis and Recognition, ICDAR05*. pp 80–84
- Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: *International joint conference on artificial intelligence (IJCAI)*, Vancouver, vol 2. <http://ijcai.org/Past%20Proceedings/IJCAI-81-VOL-2/PDF/017.pdf>, pp 674–679
- Luh JYS, Walker MW, Paul RPC (1980) On-line computational scheme for mechanical manipulators. *J Dyn Syst-T ASME* 102(2):69–76
- Lumelsky V, Stepanov A (1986) Dynamic path planning for a mobile automaton with limited information on the environment. *IEEE T Automat Contr* 31(11):1058–1063

- Luong QT (1992) matrice fondamentale et autocalibration en vision par ordinateur. Ph.D. thesis, Université de Paris-Sud, Orsay, France
- Lynch KM, Park FC (2017) Modern robotics: Mechanics, planning, and control. Cambridge University Press, New York
- Ma Y, Kosecka J, Soatto S, Sastry S (2003) An invitation to 3D. Springer-Verlag, Berlin Heidelberg
- Magnusson M, Lilienthal A, Duckett T (2007) Scan registration for autonomous mining vehicles using 3D-NDT. *J Field Robotics* 24(10):803–827
- Magnusson M, Nuchter A, Lorken C, Lilienthal AJ, Hertzberg J (2009) Evaluation of 3D registration reliability and speed – A comparison of ICP and NDT. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp 3907–3912
- Mahony R, Kumar V, Corke P (2012) Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robot Autom Mag* (19):20–32
- Maimone M, Cheng Y, Matthies L (2007) Two years of visual odometry on the Mars exploration rovers. *J Field Robotics* 24(3):169–186
- Makhlin AG (1985) Stability and sensitivity of servo vision systems. In: Proc 5th International Conference on Robot Vision and Sensory Controls – RoViSeC 5. IFS (Publications), Amsterdam, pp 79–89
- Malis E (2004) Improving vision-based control using efficient second-order minimization techniques. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp 1843–1848
- Malis E, Vargas M (2007) Deeper understanding of the homography decomposition for vision-based control. Research Report, RR-6303, Institut National de Recherche en Informatique et en Automatique (INRIA), 90 p, <https://hal.inria.fr/inria-00174036v3/document>
- Malis E, Chaumette F, Boudet S (1999) 2-1/2D visual servoing. *IEEE T Robotic Autom* 15(2):238–250
- Marey M, Chaumette F (2008) Analysis of classical and new visual servoing control laws. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Pasadena, pp 3244–3249
- Mariottini GL, Prattichizzo D (2005) EGT for multiple view geometry and visual servoing: Robotics vision with pinhole and panoramic cameras. *IEEE T Robotic Autom* 12(4):26–39
- Mariottini GL, Oriolo G, Prattichizzo D (2007) Image-based visual servoing for nonholonomic mobile robots using epipolar geometry. *IEEE T Robotic Autom* 23(1):87–100
- Marr D (2010) Vision: A computational investigation into the human representation and processing of visual information. MIT Press, Cambridge, Massachusetts
- Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. Proceedings of the 8th International Conference on Computer Vision, vol 2. pp 416–423
- Martins FN, Celeste WC, Carelli R, Sarcinelli-Filho M, Bastos-Filho TF (2008) An adaptive dynamic controller for autonomous mobile robot trajectory tracking. *Control Eng Pract* 16(11):1354–1363
- Masutani Y, Mikawa M, Maru N, Miyazaki F (1994) Visual servoing for non-holonomic mobile robots. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). Munich, pp 1133–1140
- Matić MJ (2007) The robotics primer. MIT Press, Cambridge, Massachusetts
- Matas J, Chum O, Urban M, Pajdla T (2004) Robust wide-baseline stereo from maximally stable extremal regions. *Image Vision Comput* 22(10):761–767
- Matthews ND, An PE, Harris CJ (1995) Vehicle detection and recognition for autonomous intelligent cruise control. Technical Report, University of Southampton
- Matthies L (1992) Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation. *Int J Comput Vision* 8(1):71–91
- Mayeda H, Yoshida K, Osuka K (1990) Base parameters of manipulator dynamic models. *IEEE T Robotic Autom* 6(3):312–321
- McLauchlan PF (1999) The variable state dimension filter applied to surface-based structure from motion. University of Surrey, VSSP-TR-4/99
- Merlet JP (2006) Parallel robots. Kluwer Academic, Dordrecht
- Mettler B (2003) Identification modeling and characteristics of miniature rotorcraft. Kluwer Academic, Dordrecht
- Mičušík B, Pajdla T (2003) Estimation of omnidirectional camera model from epipolar geometry. In: IEEE Conference on Computer Vision and Pattern Recognition, vol 1. Madison, pp 485–490
- Middleton RH, Goodwin GC (1988) Adaptive computed torque control for rigid link manipulations. *Syst Control Lett* 10(1):9–16
- Mikolajczyk K, Schmid C (2004) Scale and affine invariant interest point detectors. *Int J Comput Vision* 60(1):63–86
- Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. *IEEE T Pattern Anal* 27(10):1615–1630
- Mindell DA (2008) Digital Apollo. MIT Press, Cambridge, Massachusetts
- Molton N, Brady M (2000) Practical structure and motion from stereo when motion is unconstrained. *Int J Comput Vision* 39(1):5–23

- Montemerlo M, Thrun S (2007) FastSLAM: A scalable method for the simultaneous localization and mapping problem in robotics, vol 27. Springer-Verlag, Berlin Heidelberg
- Montemerlo M, Thrun S, Koller D, Wegbreit B (2002) FastSLAM: A factored solution to the simultaneous localization and mapping problem. In: Proceedings of the AAAI National Conference on Artificial Intelligence. AAAI, Edmonton, Canada
- Montemerlo M, Thrun S, Koller D, Wegbreit B (2003) FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence. Morgan Kaufmann, San Francisco, pp 1151–1156
- Moravec H (1980) Obstacle avoidance and navigation in the real world by a seeing robot rover. Ph.D. thesis, Stanford University
- Morel G, Liebezit T, Szweczyk J, Boudet S, Pot J (2000) Explicit incorporation of 2D constraints in vision based control of robot manipulators. In: Corke PI, Trevelyan J (eds) Lecture notes in control and information sciences. Experimental robotics VI, vol 250. Springer-Verlag, Berlin Heidelberg, pp 99–108
- Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. International Conference on Computer Vision Theory and Applications (VISAPP), Lisbon, Portugal (Feb 2009), pp 331–340
- Murray RM, Sastry SS, Zexiang L (1994) A mathematical introduction to robotic manipulation. CRC Press, Inc., Boca Raton
- NASA (1970) Apollo 13: Technical air-to-ground voice transcription. Test Division, Apollo Spacecraft Program Office, http://www.hq.nasa.gov/alsj/a13/AS13_TEC.PDF
- Nayar SK (1997) Catadioptric omnidirectional camera. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Los Alamitos, CA, pp 482–488
- Neilson S (2011) Robot nation: Surviving the greatest socio-economic upheaval of all time. Eridanus Press, New York, 124 p
- Neira J, Tardós JD (2001) Data association in stochastic mapping using the joint compatibility test. IEEE T Robot Autom 17(6):890–897
- Neira J, Davison A, Leonard J (2008) Guest editorial special issue on Visual SLAM. IEEE T Robot Autom 24(5):929–931
- Nethery JF, Spong MW (1994) Robotica: A mathematica package for robot analysis. IEEE T Robot Autom 1(1):13–20
- Newcombe RA, Lovegrove SJ, Davison AJ (2011) DTAM: Dense tracking and mapping in real-time. In: Proceedings of the International Conference on Computer Vision, pp 2320–2327
- Newman P (n.d.) C4B mobile robots and estimation resources. Oxford University. <http://www.robots.ox.ac.uk/~pnewman/Teaching/C4CourseResources/C4BResources.html>
- Ng J, Bräunl T (2007) Performance comparison of bug navigation algorithms. J Intell Robot Syst 50(1):73–84
- Niblack W (1985) An introduction to digital image processing. Strandberg Publishing Company Birkerød, Denmark
- Nilsson NJ (1971) Problem-solving methods in artificial intelligence. McGraw-Hill, New York
- Nistér D (2003) An efficient solution to the five-point relative pose problem. In: IEEE Conference on Computer Vision and Pattern Recognition, vol 2. Madison, pp 195–202
- Nistér D, Naroditsky O, Bergen J (2006) Visual odometry for ground vehicle applications. J Field Robotics 23(1):3–20
- Nixon MS, Aguado AS (2012) Feature extraction and image processing, 3rd ed. Academic Press, London Oxford
- Noble JA (1988) Finding corners. Image Vision Comput 6(2):121–128
- Okutomi M, Kanade T (1993) A multiple-baseline stereo. IEEE T Pattern Anal 15(4):353–363
- Ollis M, Herman H, Singh S (1999) Analysis and design of panoramic stereo vision using equi-angular pixel cameras. Robotics Institute, Carnegie Mellon University, CMU-RI-TR-99-04, Pittsburgh, PA
- Olson E (2011) AprilTag: A robust and flexible visual fiducial system. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp 3400–3407
- Orin DE, McGhee RB, Vukobratovic M, Hartoch G (1979) Kinematics and kinetic analysis of open-chain linkages utilizing Newton-Euler methods. Math Biosci 43(1/2):107–130
- Ortega R, Spong MW (1989) Adaptive motion control of rigid robots: A tutorial. Automatica 25(6):877–888
- Otsu N (1975) A threshold selection method from gray-level histograms. Automatica 11:285–296
- Papanikolopoulos NP, Khosla PK (1993) Adaptive robot visual tracking: Theory and experiments. IEEE T Automat Contr 38(3):429–445
- Papanikolopoulos NP, Khosla PK, Kanade T (1993) Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control. IEEE T Robot Autom 9(1):14–35
- Park FC (1994) Computational aspects of the product-of-exponentials formula for robot kinematics. IEEE T Automat Contr 39(3):643–647
- Paul R (1972) Modelling, trajectory calculation and servoing of a computer controlled arm. Ph.D. thesis, technical report AIM-177, Stanford University

- Paul R (1979) Manipulator Cartesian path control. *IEEE T Syst Man Cyb* 9:702–711
- Paul RP (1981) *Robot manipulators: Mathematics, programming, and control*. MIT Press, Cambridge, Massachusetts
- Paul RP, Shimano B (1978) Kinematic control equations for simple manipulators. In: *IEEE Conference on Decision and Control*, vol 17. pp 1398–1406
- Paul RP, Zhang H (1986) Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation. *Int J Robot Res* 5(2):32–44
- Piepmeyer JA, McMurray G, Lipkin H (1999) A dynamic quasi-Newton method for uncalibrated visual servoing. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Detroit, pp 1595–1600
- Pilu M (1997) A direct method for stereo correspondence based on singular value decomposition. In: *Proceedings of the Computer Vision and Pattern Recognition*, IEEE Computer Society, San Juan, pp 261–266
- Pivtoraiko M, Knepper RA, Kelly A (2009) Differentially constrained mobile robot motion planning in state lattices. *J Field Robotics* 26(3):308–333
- Pock T (2008) *Fast total variation for computer vision*. Ph.D. thesis, Graz University of Technology
- Pollefeys M, Nistér D, Frahm JM, Akbarzadeh A, Mordohai P, Clipp B, Engels C, Gallup D, Kim SJ, Merrell P, et al. (2008) Detailed real-time urban 3D reconstruction from video. *Int J Comput Vision* 78(2):143–167, Jul
- Pomerleau D, Jochem T (1995) No hands across America Journal. <http://www.cs.cmu.edu/~tjochem/nhaa/Journal.html>
- Pomerleau D, Jochem T (1996) Rapidly adapting machine vision for automated vehicle steering. *IEEE Expert* 11(1):19–27
- Posner I, Corke P, Newman P (2010) Using text-spotting to query the world. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp 3181–3186
- Pounds P (2007) *Design, construction and control of a large quadrotor micro air vehicle*. Ph.D. thesis, Australian National University
- Pounds P, Mahony R, Gresham J, Corke PI, Roberts J (2004) Towards dynamically-favourable quadrotor aerial robots. In: *Proceedings of the Australasian Conference on Robotics and Automation*. Canberra
- Pounds P, Mahony R, Corke PI (2006) A practical quad-rotor robot. In: *Proceedings of the Australasian Conference on Robotics and Automation*. Auckland
- Pounds P, Mahony R, Corke PI (2007) System identification and control of an aerobot drive system. In: *Information, Decision and Control*. IEEE, pp 154–159
- Poynton CA (2003) *Digital video and HDTV: Algorithms and interfaces*. Morgan Kaufmann, San Francisco
- Poynton CA (2012) *Digital video and HD algorithms and interfaces*. Morgan Kaufmann, Burlington
- Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2007) *Numerical recipes*, 3rd ed. Cambridge University Press, New York
- Prince SJ (2012) *Computer vision: Models, learning, and inference*. Cambridge University Press, New York
- Prouty RW (2002) *Helicopter performance, stability, and control*. Krieger, Malabar FL
- Pyncheon T (2006) *Against the day*. Jonathan Cape, London
- Rekleitis IM (2004) A particle filter tutorial for mobile robot localization. Technical report (TR-CIM-04-02), Centre for Intelligent Machines, McGill University
- Rives P, Chaumette F, Espiau B (1989) Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In: Hayward V, Khatib O (eds) *Lecture notes in control and information sciences*. Experimental robotics I, vol 139. Springer-Verlag, Berlin Heidelberg, pp 412–428
- Rizzi AA, Koditschek DE (1991) Preliminary experiments in spatial robot juggling. In: Chatila R, Hirzinger G (eds) *Lecture notes in control and information sciences*. Experimental robotics II, vol 190. Springer-Verlag, Berlin Heidelberg, pp 282–298
- Roberts LG (1963) *Machine perception of three-dimensional solids*. MIT Lincoln Laboratory, TR 315, <http://www.packet.cc/files/mach-per-3D-solids.html>
- Rosenfield GH (1959) The problem of exterior orientation in photogrammetry. *Photogramm Eng* 25(4):536–553
- Rosten E, Porter R, Drummond T (2010) FASTER and better: A machine learning approach to corner detection. *IEEE T Pattern Anal* 32:105–119
- Russell S, Norvig P (2009) *Artificial intelligence: A modern approach*, 3rd ed. Prentice Hall Press, Upper Saddle River, NJ
- Sakaguchi T, Fujita M, Watanabe H, Miyazaki F (1993) Motion planning and control for a robot performer. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Atlanta, May, pp 925–931
- Salvi J, Matabosch C, Fofi D, Forest J (2007) A review of recent range image registration methods with accuracy evaluation. *Image Vision Comput* 25(5):578–596

- Samson C, Espiau B, Le Borgne M (1990) Robot control: The task function approach. Oxford University Press, Oxford
- Sanderson AC, Weiss LE, Neuman CP (1987) Dynamic sensor-based control of robots with visual feedback. *IEEE T Robotic Autom* RA-3(5):404–417
- Scaramuzza D, Fraundorfer F (2011) Visual odometry [tutorial]. *IEEE Robot Autom Mag* 18(4):80–92
- Scharstein D, Pal C (2007) Learning conditional random fields for stereo. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*. Minneapolis, MN
- Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int J Comput Vision* 47(1):7–42
- Selig JM (2005) *Geometric fundamentals of robotics*. Springer-Verlag, Berlin Heidelberg
- Sharp A (1896) *Bicycles & tricycles: An elementary treatise on their design and construction; With examples and tables*. Longmans, Green and Co., London New York Bombay
- Sheridan TB (2003) *Telerobotics, automation, and human supervisory control*. MIT Press, Cambridge, Massachusetts, 415 p
- Shi J, Tomasi C (1994) Good features to track. In: *Proceedings of the Computer Vision and Pattern Recognition*. IEEE Computer Society, Seattle, pp 593–593
- Shih FY (2009) *Image processing and mathematical morphology: Fundamentals and applications*. CRC Press, Boca Raton
- Shirai Y (1987) *Three-dimensional computer vision*. Springer-Verlag, New York
- Shirai Y, Inoue H (1973) Guiding a robot by visual feedback in assembling tasks. *Pattern Recogn* 5(2):99–106
- Shoemaker K (1985) Animating rotation with quaternion curves. In: *Proceedings of ACM SIGGRAPH*, San Francisco, pp 245–254
- Siciliano B, Khatib O (eds) (2016) *Springer handbook of robotics*, 2nd ed. Springer-Verlag, New York
- Siciliano B, Sciavicco L, Villani L, Oriolo G (2009) *Robotics: Modelling, planning and control*. Springer-Verlag, Berlin Heidelberg
- Siegwart R, Nourbakhsh IR, Scaramuzza D (2011) *Introduction to autonomous mobile robots*, 2nd ed. MIT Press, Cambridge, Massachusetts
- Silver WM (1982) On the equivalence of Lagrangian and Newton-Euler dynamics for manipulators. *Int J Robot Res* 1(2):60–70
- Sivic J, Zisserman A (2003) Video Google: A text retrieval approach to object matching in videos. In: *Proceedings of the Ninth IEEE International Conference on Computer Vision*. pp 1 470–1 477
- Skaar SB, Brockman WH, Hanson R (1987) Camera-space manipulation. *Int J Robot Res* 6(4):20–32
- Skofteland G, Hirzinger G (1991) Computing position and orientation of a freeflying polyhedron from 3D data. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Seoul, pp 150–155
- Slama CC (ed) (1980) *Manual of photogrammetry*, 4th ed. American Society of Photogrammetry
- Smith R (2007) An overview of the Tesseract OCR engine. In: *9th International Conference on Document Analysis and Recognition (ICDAR)*. pp 629–633
- Sobel D (1996) *Longitude: The true story of a lone genius who solved the greatest scientific problem of his time*. Fourth Estate, London
- Soille P (2003) *Morphological image analysis: Principles and applications*. Springer-Verlag, Berlin Heidelberg
- Spong MW (1989) Adaptive control of flexible joint manipulators. *Syst Control Lett* 13(1):15–21
- Spong MW, Hutchinson S, Vidyasagar M (2006) *Robot modeling and control*, 2nd ed. John Wiley & Sons, Inc., Chichester
- Srinivasan VV, Venkatesh S (1997) *From living eyes to seeing machines*. Oxford University Press, Oxford
- Stachniss C, Burgard W (2014) Particle filters for robot navigation. *Foundations and Trends in Robotics* 3(4):211–282
- Steinval A (2002) *English colour terms in context*. Ph.D. thesis, Ume Universitet
- Stentz A (1994) The D* algorithm for real-time planning of optimal traverses. The Robotics Institute, Carnegie-Mellon University, CMU-RI-TR-94-37
- Stewart A (2014) *Localisation using the appearance of prior structure*. Ph.D. thesis, University of Oxford
- Stone JV (2012) *Vision and brain: How we perceive the world*. MIT Press, Cambridge, Massachusetts
- Strasdat H (2012) *Local accuracy and global consistency for efficient visual SLAM*. Ph.D. thesis, Imperial College London
- Strelow D, Singh S (2004) Motion estimation from image and inertial measurements. *Int J Robot Res* 23(12):1 157–1 195
- Sünderhauf N (2012) *Robust optimization for simultaneous localization and mapping*. Ph.D. thesis, Technische Universität Chemnitz
- Sussman GJ, Wisdom J, Mayer ME (2001) *Structure and interpretation of classical mechanics*. MIT Press, Cambridge, Massachusetts
- Sutherland IE (1974) Three-dimensional data input by tablet. *P IEEE* 62(4):453–461

- Svoboda T, Pajdla T (2002) Epipolar geometry for central catadioptric cameras. *Int J Comput Vision* 49(1):23–37
- Szeliski R (2011) *Computer vision: Algorithms and applications*. Springer-Verlag, Berlin Heidelberg
- Tahri O, Chaumette F (2005) Point-based and region-based image moments for visual servoing of planar objects. *IEEE T Robot Autom* 21(6):1116–1127
- Tahri O, Mezouar Y, Chaumette F, Corke PI (2009) Generic decoupled image-based visual servoing for cameras obeying the unified projection model. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe, pp 1116–1121
- Taylor RA (1979) Planning and execution of straight line manipulator trajectories. *IBM J Res Dev* 23(4):424–436
- ter Haar Romeny BM (1996) *Introduction to scale-space theory: Multiscale geometric image analysis*. Utrecht University
- Thrun S, Burgard W, Fox D (2005) *Probabilistic robotics*. MIT Press, Cambridge, Massachusetts
- Tissainayagam P, Suter D (2004) Assessing the performance of corner detectors for point feature tracking applications. *Image Vision Comput* 22(8):663–679
- Titterton DH, Weston JL (2005) *Strapdown inertial navigation technology*. IEE Radar, Sonar, Navigation and Avionics Series, vol 17, The Institution of Engineering and Technology (IET), 576 p
- Tomasi C, Kanade T (1991) Detection and tracking of point features. Carnegie Mellon University, CMU-CS-91-132
- Triggs B, McLauchlan P, Hartley R, Fitzgibbon A (2000) Bundle adjustment – A modern synthesis. *Lecture notes in computer science. Vision algorithms: theory and practice*, vol 1883. Springer-Verlag, Berlin Heidelberg, pp 153–177
- Tsakiris D, Rives P, Samson C (1998) Extending visual servoing techniques to nonholonomic mobile robots. In: Kriegman DJ, Hager GD, Morse AS (eds) *Lecture notes in control and information sciences. The confluence of vision and control*, vol 237. Springer-Verlag, Berlin Heidelberg, pp 106–117
- Uicker JJ (1965) On the dynamic analysis of spatial linkages using 4 by 4 matrices. Dept. Mechanical Engineering and Astronautical Sciences, NorthWestern University
- Usher K (2005) Visual homing for a car-like vehicle. Ph.D. thesis, Queensland University of Technology
- Usher K, Ridley P, Corke PI (2003) Visual servoing of a car-like vehicle – An application of omnidirectional vision. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Taipei, Sep, pp 4288–4293
- Valgren C, Lilienthal AJ (2010) SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments. *Robot Auton Syst* 58(2):149–156
- Vanderborcht B, Sugar T, Lefeber D (2008) Adaptable compliance or variable stiffness for robotic applications. *IEEE Robot Autom Mag* 15(3):8–9
- Vedaldi A, Fulkerson B (2008) VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org>
- Wade NJ (2007) Image, eye, and retina. *J Opt Soc Am A* 24(5):1229–1249
- Walker MW, Orin DE (1982) Efficient dynamic computer simulation of robotic mechanisms. *J Dyn Syst-T ASME* 104(3):205–211
- Walter WG (1950) An imitation of life. *Sci Am* 182(5):42–45
- Walter WG (1951) A machine that learns. *Sci Am* 185(2):60–63
- Walter WG (1953) *The living brain*. Duckworth, London
- Warren M (2015) Long-range stereo visual odometry for unmanned aerial vehicles. Ph.D. thesis, Queensland University of Technology
- Weiss LE (1984) Dynamic visual servo control of robots: An adaptive image-based approach. Ph.D. thesis, technical report CMU-RI-TR-84-16, Carnegie-Mellon University
- Weiss L, Sanderson AC, Neuman CP (1987) Dynamic sensor-based control of robots with visual feedback. *IEEE T Robot Autom* 3(1):404–417
- Westmore DB, Wilson WJ (1991) Direct dynamic control of a robot using an end-point mounted camera and Kalman filter position estimation. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Seoul, Apr, pp 2376–2384
- Whitney DE (1969) Resolved motion rate control of manipulators and human prostheses. *IEEE T Man Machine* 10(2):47–53
- Wiener N (1965) *Cybernetics or control and communication in the animal and the machine*. MIT Press, Cambridge, Massachusetts
- Wilburn B, Joshi N, Vaish V, Talvala E-V, Antunez E, Barth A, Adams A, Horowitz M, Levoy M (2005) High performance imaging using large camera arrays. *ACM Transactions on Graphics (TOG) – Proceedings of ACM SIGGRAPH 2005* 24(3):765–776
- Wolf PR (1974) *Elements of photogrammetry*. McGraw-Hill, New York
- Woodfill J, Von Herzen B (1997) Real-time stereo vision on the PARTS reconfigurable computer. In: *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines*, Grenoble. pp 201–210
- Xu G, Zhang Z (1996) *Epipolar geometry in stereo, motion, and object recognition: A unified approach*. Springer-Verlag, Berlin Heidelberg

- Ying X, Hu Z (2004) Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model. In: Pajdla T, Matas J (eds) Lecture notes in computer science. Computer vision – ECCV 2004, vol 3 021. Springer-Verlag, Berlin Heidelberg, pp 442–455
- Yoshikawa T (1984) Analysis and control of robot manipulators with redundancy. In: Brady M, Paul R (eds) Robotics research: The first international symposium. MIT Press, Cambridge, Massachusetts, pp 735–747
- Zabih R, Woodfill J (1994) Non-parametric local transforms for computing visual correspondence. In: Ecklundh J-O (ed) Lecture notes in computer science. Computer Vision – ECCV 1994, vol 800. Springer-Verlag, Berlin Heidelberg, pp 151–158
- Zarchan P, Musoff H (2005) Fundamentals of Kalman filtering: A practical approach. Progress in Astronautics and Aeronautics, vol 208. American Institute of Aeronautics and Astronautics
- Zhang Z, Faugeras O, Kohonen T, Hunag TS, Schroeder MR (1992) Three D-dynamic scene analysis: A stereo based approach. Springer-Verlag, New York
- Ziegler J, Bender P, Schreiber M, Latégahn H, Strauss T, Stiller C, Thao Dang, Franke U, Appenrodt N, Keller CG, Kaus E, Herrtwich RG, Rabe C, Pfeiffer D, Lindner F, Stein F, Erbs F, Enzweiler M, Knöppel C, Hipp J, Haueis M, Trepte M, Brenk C, Tamke A, Ghanaat M, Braun M, Joos A, Fritz H, Mock H, Hein M, Zeeb E (2014) Making Bertha drive – An autonomous journey on a historic route. IEEE Intelligent Transportation Systems Magazine 6(2):8–20

Index

Index of People

A

Ackermann, Rudolph 101
Alhazen, aka 331
Asimov, Issac 1

B

Ball, Sir Robert 52, 55
Bayer, Bryce E. 294
Bayes, Reverend Thomas 157
Beer, August 289
Black, Harold 4
Bode, Henrik 4
Boltzman, Ludwig 288
Braitenberg, Valentino 126
Bryan, George 37

C

Čapek, Karel 1, 3
Cardano, Gerolamo 37
Chasles, Michel 53
Chrétien, Henri 366
Clifford, William 55
Cook, Captain James 152, 167
Coriolis, Gaspard-Gustave de 69
Coulomb, Charles-Augustin de 253

D

Dalton, John 295
Davy, Sir Humphry 289
Delaunay, Boris 137
Denavit, Jacques 198
Descartes, René 19
Devol, George C. Jr. 1, 2
Draper, Charles Stark (Doc) 81, 82, 158

E

Edison, Thomas Alva 289
Einstein, Albert 70
Engelberger, Joseph F. 2
Euclid of Alexandria 18, 331
Euler, Leonhard 36, 68, 265

G

Gauss, Carl Friedrich 61, 383

Goetz, Raymond 7

H

Hall, Edwin 85
Hamilton, Sir William Rowan 44, 55, 60, 61
Harrison, John 152
Hartenberg, Richard 198
Helmholtz, Hermann, von 293
Hering, Karl Ewald 293
Herschel, William 288, 337
Hershey, Allen V. 220
Hesse, Ludwig Otto 618
Hough, Paul 455

I

Ilon, Bengt 112

J

Jacobi, Carl Gustav Jacob 232

K

Kálmán, Rudolf 157
Kepler, Johannes 331

L

Lagrange, Joseph-Louis 265
Lambert, Johann Heinrich 309
Land, Edward 307
Laplace, Pierre-Simon 383, 384
Laussedat, Aimé 354
Lazzarini, Mario 174
Leclerc, Georges-Louis 174
Lie, Sophus 611

M

Markov, Andrey 137
Marr, David 388
Maxwell, James Clerk 293
McCarthy, John 4
McCulloch, Warren 4
Metropolis, Nicholas 174
Minsky, Marvin 4
Moler, Cleve 9
Mozi (Chinese philosopher of 5th century BCE) 319

N

Newell, Allen 4
 Newton, Sir Isaac 67–69, 279, 287, 337
 Nyquist, Harold 4, 403

P

Pitts, Walter 4
 Planck, Max 288
 Plücker, Julius 597
 Price, Richard 157
 Ptolemy, Claudius 331

R

Rodrigues, Olinde 42, 61

S

Scheinman, Victor 195
 Schmidt, Stanley F. 158
 Shannon, Claude 4, 403
 Simon, Herbert 4
 Sobel, Irwin 407
 Stefan, Jozef 288
 Swan, Sir Joseph 289

Index of Functions, Classes and Methods

Classes are shown in **bold**, Simulink® models in *italics*, and methods are prefixed by a dot. All others are Toolbox functions.

A

abcolorspace 421
 about 77, 118, 145, 204, 213, 264, 360–362, 365, 378, 402, 427, 485, 520, 551, 573, 620
 abs 372
AlphaBeta 579
 anaglyph 495
 angdiff 103, 566, 577
 angles 48
 angvec2r 42
 angvec2tr 61
 animate 76, 77
AprilTag 164
 apriltags 164, 184
 atan2 372, 601
AxisWebCamera 366
 →, .grab 366
 →, .size 366

B

BagOfWords 515, 517, 519
 →, .contains 516
 →, .exemplars 516
 →, .occurrence 516
 →, .remove_stop 516
 →, .similarity 517, 519
 →, .wordfreq 516
 →, .words 515

T

Tait, Peter 37, 61
 Tesla, Nikola 6
 Turing, Alan 4

U

Ulam, Stanislaw 174

V

von Goethe, Johann Wolfgang 293
 von Kármán, Theodore 198
 von Neumann, John 174
 Voronoy, Georgy Feodosevich 137

W

Wald, George 292
 Walter, William Grey 4, 126
 Wien, Wilhelm 288
 Wiener, Norbert 4

Y

Young, Thomas 293

→, .wordvector 517
Bicycle 100, 101, 109, 111
Bicycle 163, 166, 168, 176, 185, 575
 BinaryOccupancyGrid 148
 blackbody 288, 289, 300, 306, 307
 boundary 434
 boundmatch 436
 bug 129
 bug.plot 129
bug2 129
 →, .path 129, 131
BundleAdjust 498, 499, 502, 641
 →, .addcamera 498
 →, .errors 501
 →, .getcamera 502
 →, .getstate 500
 →, .optimize 501
 →, .plot 499, 502
 bwdist 399

C

calib_gui_normal 336
 cam 323–325, 481
 →, .estpose 539
 →, .grab 364
 camcald 354
Camera 327, 338, 341, 343, 355, 551
Camera2 579
 cast 373

CatadioptricCamera 341
ccode 620
CentralCamera 323, 331, 338, 341, 343, 352, 355, 465, 467, 468,
 473, 477, 479, 501, 539, 542, 544, 548, 549, 569, 573, 607
 → .C 326
 → .clf 356
 → .derivs 501
 → .E 468
 → .estpose 539
 → .F 467
 → .f 546
 → .flowfield 544, 546
 → .fov 327
 → .hold 356
 → .invC 327
 → .invE 481
 → .invH 477, 511
 → .K 557
 → .mesh 329, 339, 341, 356
 → .move 469, 481, 482, 497
 → .plot 328, 329, 356, 473, 539, 549
 → .plot_epiline 468, 473, 479
 → .pp 549
 → .project 323–325, 328, 335, 497, 542, 557, 607
 → .ray 482
 → .T 329
 → .visjac_e 558
 → .visjac_p 544, 546
 → .visjac_p_polar 568
children 432
circle 556, 557
clf 356
closest 505
cmfrgb 297, 298
cmfxyz 300
colnorm 476, 497
colorkmeans 420–422
colorname 301, 305, 421, 423, 491
colorspace 302, 303, 312
cones 292, 293
ctray 8, 78, 214, 215
cylinder 329, 356

D

delta2tr 67, 549
descriptor 461
DHFactor 218, 222
 → .dh.command 222
diff 381, 523
DigitalCamera 480
dim 163
distance 461
Dstar 134
 → .costmap 134
 → .modify_cost 135
 → .miter 135
 → .path 135
 → .plan 134, 135
DXform 131
 → .path 131
 → .plan 131, 132
 → .visualize 131
 → .visualize3d 132

E

e2h 29, 467
EarthView 367, 410
 → .grab 367
ebundle 523
efund 523
eig 41, 430, 504
EKF 158, 160, 163, 166–168
ellipsoid 329
epidist 471
eps 50
ETS2 194, 227
 → .fkine 194
 → .plot 195
 → .Rz 194, 196
 → .structure 195, 196
 → .teach 194
 → .Tx 194, 196
ETS3 227
 → .fkine 196
 → .Ry 196
 → .Rz 196
 → .Tx 196
 → .Ty 196
 → .Tz 196
eul2jac 233
eul2r 36, 37
eul2tr 48
eval 218, 222
ExampleHelperRobot Simulator 123
exp 26, 43
expm 26, 43, 47, 52, 54, 61
eye 29, 218

F

fcode 620
FeatureMatch 463, 472, 478, 496
 → .inlier 473, 478, 479, 482
 → .outlier 473, 478
 → .plot 463, 473
 → .ransac 472
 → .show 472, 478
 → .subset 463, 473, 479, 482
FishEyeCamera 338, 339, 355
fkine 214
flowers 302
fmatrix 470, 471, 531
fminsearch 207

G

gait 225
gaussfunc 631, 632
GeometricJacobian 248
ginput 531
Graph 214

H

h2e 29, 467, 468, 473
histogram 463
homography 474, 510, 531

homtrans 470, 474–476, 557
homwarp 510, 513
Hough 441, 442
 -, .lines 441, 442
 -, .plot 442
 -, .show 441
humoments 434

I

animate 520
ibox 428
iblobs 432–434
IBVS 549
 -, .plot_camera 569
 -, .plot_error 569
 -, .plot_vel 569
 -, .step 549
IBVS_polar 569
IBVS_sph 571
icanny 385, 440, 442
icensus 531
iclose 397
icolor 370, 373
iconv 377, 378, 382–384, 387
icorner 445, 446, 448, 460, 461, 520
icp 180
idecimate 403
idisp 311, 314, 346, 347, 361, 362, 367, 368, 372–374,
 376, 378, 382, 383, 390, 392–394, 398, 401, 410,
 415, 418, 419, 421, 424–426, 434, 442, 445, 446,
 448–451, 453, 460, 463, 473, 478, 484, 487, 490,
 491, 509, 515, 517
idouble 362, 370, 373, 405
iendpoint 399
igamma 372, 373
igraphcut 457
ihist 369, 372, 373, 416, 487
iint 370
ImageSource 365
imeshgrid 404–406, 493
imoments 431, 432
imono 362, 370
imorph 394, 395, 397, 398
imser 419
InitFcn 275, 573, 577
inormhist 372
interp 77, 78
interp2 405
invariant 314, 315
invcamcal 333, 334
InverseKinematics 227
iopen 397, 421
ipaste 368, 512, 513
ipixswitch 374, 487, 492, 494
ipyramid 403
irank 397, 531
iread 314, 345, 360–363, 373, 378, 390, 396, 401, 402, 406,
 415, 417, 419, 426, 433, 437, 438, 442, 445, 448, 449,
 451, 460, 478, 480, 483, 494, 509, 510, 512, 514, 518, 520
irectify 496
ireplicate 403
iroi 388, 401, 410
irotate 405, 440
isamesize 374

iscale 403
iscalemax 449
iscalespace 449, 451
isift 456
isimilarity 390, 392, 410
ismooth 378, 403
isobel 384
isosurface 357
istereio 484, 485, 487, 490, 496, 531
istretch 372
isurf 460, 462, 478, 496, 512, 514, 518
ithin 136, 399
ithresh 416
itriplepoint 399
iwindow 397

J

jacobian 172, 230, 620
Jacobian 237, 551
jsingu 234
jtraj 204, 212, 214, 216, 263, 274, 275

K

kcircle 368, 379, 397, 398, 421
kdgauss 384
kdog 385
kgauss 378, 384
klog 387
kmeans 423

L

lambda2rg 298, 299
lambda2xy 300, 306, 307, 309
LandmarkMap 160, 163
Lattice 141, 641
LineFeature 441, 442
 -, .plot 442
 -, .seglength 442
lines 442
Link 198, 200, 217, 256, 268
 -, .A 199
 -, .a 199
 -, .offset 199
 -, .RP 199
loadspectrum 289–291, 307, 309, 312, 317
log 25, 42
logm 25, 42, 54
lscov 248
lspb 72, 73, 78, 212
luminos 291, 292

M

makemap 131, 148
matlabFunction 620
max 372, 376, 492, 645
mdl_puma560 201, 207, 263, 265, 274
mdl_quadrotor 118
mdl_twolink 253
meshgrid 265, 266, 345, 347, 357
min 372, 376, 492
mkcube 329, 332, 334, 339, 341, 343

mkgrid 474, 504, 539, 548
model 103
 models 200
MonteCarloLocalization 185
Movie 365, 366, 375
 -, .framerate 365
 -, .grab 365, 375, 376
 -, .nframes 365
 -, .size 365
 -, .skiptoframe 365
 -, .skiptotime 365
 mplot 238
 mpq 428, 429
 mpq_point 603
 mpq_poly 435, 566
 mstraj 220, 224
 mtraj 73, 76, 212
 mxn 393

N

Navigation 130
 ncc 388, 389, 462
 niblack 418
 npq 434
 npq_poly 435
 null 468, 622
 numcols 81, 89, 90, 139, 220, 265, 266, 384, 497, 558, 603
 numrows 131, 220, 384

O

oa2r 40
 ocr 455
 otsu 417

P

ParticleFilter 177, 185
 pathtool 583
PBVS 539
 pcregrigid 183
 peak 369, 645–647
 peak2 390, 647
PGraph 499, 641
 -, .add_edge 641, 642
 -, .add_node 641
 -, .closest 642
 -, .cost 642
 -, .edges 642
 -, .neighbours 642
 -, .plot 642
 pinv 240, 241, 243, 549
ploop 261, 280
ploop_test 261
 plot 179
 plot_box 428, 432, 437
 plot_circle 390
 plot_frame 334
 plot_homline 351, 606
 plot_point 28, 390, 478
 plot_poly 509
 plot_sphere 334, 466, 475
 plot2 299, 551
 plotbox 437

Plucker 54, 351, 607
 -, .L 607
 -, .side 597
 -, .w 351
 pnmfilt 409
PointFeature 445, 446, 448, 449, 460, 461, 520
 -, .descriptor 461
 -, .plot 446, 448, 460
 pol2cart 179
Polygon 149
PoseGraph 172, 173, 179, 641
 -, .optimize 173
 -, .plot 172, 173
 -, .plotccgrid 181
 -, .scan 179
 -, .scanmap 181
 -, .scanxy 179
 -, .time 180
PRM 138, 148, 641
 -, .path 139
 -, .plan 138
 -, .visualize 139

Q

q.animate 76
 qplot 213
 qr 327
Quaternion 44

R

rand 139
 randinit 138, 139
 randn 139, 631
RandomPath 157, 166, 176
RangeBearingSensor 161, 162, 164, 166–168, 177, 186
 -, .h 162
 -, .H_w 162
 -, .H_x 162
 -, .reading 161
 ransac 471, 472, 476, 496, 512
 ransac_driver 531
Ray3D 481, 482
 -, .intersect 482
RegionFeature 431–434
 -, .boundary 435
 -, .boundmatch 436
 -, .moments 431
 -, .plot_boundary 434
 -, .plot_box 432, 434
 -, .plot_centroid 432
 -, .plot_ellipse 432
 -, .shape 431
 -, .theta 431
 -, .uc 431
Revolute 198
RevoluteMDH 219
 rgb2hsv 302
RigidBodyTree 227, 248
RNE 274, 280
roblocks 103, 111
Robot 271
 rotx 34, 35, 42, 43, 66
 roty 34–36

rotz 34, 36
 rpy2r 41, 62
 rpy2tr 45, 332, 504
 RRT 144, 641
 → .path 145
 → .plan 144
 → .visualize 144
 running 103

S

sad 388, 389
ScalePointFeature 449, 452
 se2 405
 SE2 57, 172, 201
 → .Rx 211
 se2 405
 SE3 57, 203, 204, 208, 210, 211, 214–216, 221, 224, 233
 → .Rx 216
 → .Ry 215
 → .Rz 224
 → .torotvec 233
Sensor 161, 166
 → .H_xf 166
 sensorfield 127, 148
 seq 362
SerialLink 200, 206, 212, 224, 227, 249, 265
 → .accel 271
 → .base 203, 204, 265
 → .coriolis 263
 → .edit 200
 → .fdyn 272
 → .fellipse 245
 → .fkine 200, 203, 204, 207, 209, 213, 230, 238
 → .gravity 265
 → .gravload 264, 265, 269
 → .ikcon 227
 → .ikine 208, 210, 215, 216, 224, 227, 246
 → .ikine6s 207–211, 214–216, 221, 227
 → .ikinesym 206
 → .inertia 263, 266, 268, 270
 → .jacob0 231–236, 244, 270
 → .jacobn 232
 → .jtraj 212
 → .links 269
 → .maniply 215, 236, 271
 → .motordynamics 256
 → .nofriction 272
 → .plot 203, 209, 213, 216, 221, 227, 271
 → .plot3d 227
 → .rne 263, 264, 269
 → .teach 227, 235, 245, 248
 → .tool 203
 → .vellipse 235
 shortest 76
 showcolorspace 300, 312
SiftPointFeature 456
 sigma 631
 simplify 25, 62, 172
 skew 613
 sl_arm_ibvs 572, 573
 sl_bicycle 101
 sl_braitenberg 126, 127
 sl_ctorque 273–275
 sl_driveline 104, 105
 sl_drivepoint 103, 104
 sl_drivepose 108, 109
 sl_drivepose_vs 575, 576
 sl_fforward 273, 274
 sl_ibvs 550, 551, 575
 sl_jspace 214, 215
 sl_lanechange 101, 102
 sl_mobile_vs 574
 sl_omni_vs 574
 sl_opspace 275
 sl_partitioned 566, 567
 sl_pursuit 106
 sl_quadcopter 118
 sl_quadcopter_vs 577
 sl_quadrotor 117, 118
 sl_quadrotor_vs 576, 577
 sl_rrmc 237, 238
 sl_rrmc2 238, 239
 sl_sea 277
 sl_ztorque 271
 SO2 57, 74
 SO3 57, 76, 504
 sol 469
 sphere 329, 356
SphericalCamera 343, 355, 571, 576
 → .grab 364
 → .mesh 343
 → .size 364
 spy 629
 sqrt 159, 270, 372, 384, 406, 430, 502
 sqrtm 601
 ssd 388, 389
 stdisp 483, 484, 496, 497
 stereo 484
SurfPointFeature 452, 460, 462, 478, 496, 512, 514, 518
 → .match 463, 464, 478, 479, 496, 512
 → .plot_scale 453, 460
 → .scale 453
 → .support 515

T

T1.torpy 73
 t2r 47
 t2rt 481
 T2xyz 214
 tags 164
 tau-d 259
 tau_ff 260
 testpattern 367, 440
Tgrid 474, 477
 tic 410
 toc 410
 torpy 77
 tpoly 71, 72, 78, 91, 92, 212
 tr2angvec 41
 tr2delta 67
 tr2eul 36, 37
 tr2rotvec 233
 tr2rpy 38, 511
 Tracking Controller 111
 traj 220
 tranimate 34, 35, 61, 62
 transl 8, 47, 203, 204, 208, 210, 213, 215, 223, 224, 238, 324, 329,
 332, 465, 470, 474, 504, 539, 542, 548, 549, 554, 569, 571, 573

transl2 27, 28
 trexp 43, 52, 54, 61
 trinterp 78
 tripleangle 38, 62
 triplepoint 136
 tristem2cc 299, 305, 373
 trlog 43, 54, 233
 trnorm 539, 549
 trot2 27
 trotx 47, 48, 204, 222, 265, 329, 474, 542, 573
 troty 215, 329, 465, 474
 trotz 222, 329, 539, 549, 554, 569, 571
 trplot 27, 35, 47, 61
 trplot2 28, 61
Ts
 -, .t 214
 -, .torpy 214
Twist 30, 54, 201
 -, .expm 54
 -, .line 54
 -, .S 54
 -, .T 30, 54, 201

U

uint8 302, 360–362
Unicycle 111, 123, 185
UnitQuaternion 45, 46, 50, 68, 76, 81
 -, .animate 81
 -, .dot 64

General Index

Symbols

\-operator 71, 558, 621, 623
 3D reconstruction 350, 459, 492, 527

A

A* search 134, 139, 142, 643
aberration
 -, chromatic 330
 -, spherical 330
absorption 289, 309
 -, coefficient 290
 -, color change 308
 -, light, differential 290
 -, shock 277
 -, spectrum 290, 309
 -, water 289, 309
acceleration 81, 82, 87, 120, 251, 275
 -, angular 68
 -, centripetal 70
 -, Coriolis 70, 91
 -, discontinuity 78
 -, Euler 70
 -, gravitational 70, 83
 -, inertial 83
 -, proper 83
 -, sensor 83, 87
 -, specific 83
accelerometer 39, 41, 81–83, 87
 -, triaxial 83, 87

 -, .dotb 64
 -, .omega 81
 -, .plot 68
 -, .torpy 81
upq 429
upq_poly 435

V

Vehicle 156, 158, 160, 185
 -, .Fv 158
 -, .Fx 158
 -, .step 157
vex 25, 26, 42, 43, 613
VideoCamera 363, 365, 366
 -, .grab 364
 -, .size 364
VisualServo 541, 549
vl_imdisttf 399
vloop 257, 280
vloop_test 258
VREP_class 187

X, Y, Z

xv 620
zcross 387
znc 389
zsd 389
zssd 389

acomodation 321
ACF (see *aggregate channel feature*)
Ackermann steering 101
actuation 120
 -, electric 256
 -, electro-hydraulic 251
actuator 120, 251
 -, joint 252
 -, saturation 118
 -, series-elastic (SEA) 276, 277
addition
 -, Minkowski 395
 -, vector 587
adjoint
 -, logarithm of 615
 -, matrix 65, 69, 201, 247, 597, 615
adjugate 352, 589, 607
adjustment, bundle 184, 497–503, 521–523, 527
affine
 -, camera 353, 503
 -, reconstruction 503
 -, space 608
 -, transformation 608
AGAST detector 462
aggregate channel feature (ACF) descriptor 462
AHRS (see *attitude and heading reference system*)
aircraft 119, 121
Airy pattern 378
albedo 290
algebra 611
algebraic group 612

- algorithm
 - , box filter aggregation 526
 - , Bresenham 181
 - , *bug* 128–130
 - , clustering 455
 - , D* 134
 - , dense stereo matching 526
 - , FastSLAM (see also *Rao-Blackwellized SLAM*) 183
 - , graphcuts 438
 - , hybrid visual-servo 536
 - , ICP (iterated closest point) 183, 506, 526
 - , *k*-means 420
 - , Levenberg-Marquardt 246, 624, 625
 - , MSER (maximally stable extremal region) 419, 438, 454, 462
 - , NCC matching (similarity measure) 388, 389, 410, 526, 531
 - , Newton-Euler 263
 - , Niblack 418
 - , pose estimation 537
 - , RANSAC (random sampling and consensus) 471, 472, 476, 478, 504
 - , rapidly exploring 145
 - , resolved-rate motion control 237
 - , RRT (rapidly-exploring random tree) 145
 - , SGM (semi-global matching) 526
 - , skeletonization 136, 137
 - , stereo matching 486
 - , subpixel refinement 526
 - , SURF (speeded up robust feature) 453, 462, 463, 472, 478, 479, 496, 514–516, 524, 556
 - , thinning 136, 137
 - , velocity loop control 257
 - , winner takes all 526
 - aliasing
 - , anti- 402, 407
 - , spatial 402, 486, 488
 - ambiguity ratio 486, 530
 - ampullae 83
 - anaglyph 495
 - , image 495
 - , stereo glasses 35
 - analysis
 - , blob 455
 - , connected component 424, 425, 438
 - , image
 - , segmentation 455
 - , sequence 527
 - , root-locus 280
 - analytical Jacobian 233
 - anamorphic lens 366
 - angle
 - , Cardan 32, 38
 - , declination 85
 - , elevation 152
 - , Euler 36, 37, 38, 40, 59, 75, 196, 232, 233, 247, 499, 571
 - , singularity 39
 - , heading 87
 - , inclination 85
 - , joint 5, 13, 198
 - , nautical 38
 - , representation 36
 - , roll-pitch-yaw 37, 38, 78, 212–214, 232
 - , rate 76
 - , singularity 38
 - , rotation 25, 26, 31, 35, 37, 39, 43
 - , solid 288, 294, 326
 - , steering 101, 102, 141, 145
 - , Tait-Bryan 38
 - , trajectory
 - , joint 272
 - , LSPB (linear segment with parabolic blend) 72, 261, 262
 - , XYZ sequence 38
 - angle-axis representation 41, 45, 499
 - angular
 - , acceleration 68
 - , momentum 68, 79, 80
 - , rate 88
 - , uncertainty 159
 - , velocity 50, 52, 64, 68, 70, 79, 80, 155, 233, 636
 - anthropomorphic 147, 202, 203
 - anti-aliasing 368, 402, 405, 407
 - anti-symmetric matrix 589
 - aperture 349, 364
 - , lens 321, 331
 - Apollo
 - , 13 38, 40
 - , Lunar Module 39, 81
 - approach vector 40, 41, 210, 211
 - April tag 164
 - architecture, subsumption 127
 - ArduCopter (software project) 122
 - artificial intelligence 4
 - Asimo humanoid robot 6
 - aspect ratio 324, 366, 430–433
 - astigmatism 330
 - Asus Xtion 508
 - ASV (see *autonomous surface vehicle*)
 - attitude and heading reference system (AHRS) 87
 - autocorrelation matrix 444
 - automata 128
 - automated guided vehicle 96
 - autonomous surface vehicle (ASV) 96
 - autonomous underwater vehicle (AUV) 96, 120, 121
 - axis
 - , instantaneous 64
 - , of motion 73
 - , optical 40, 321, 325, 496, 509, 510, 541, 554, 566, 568, 570
 - , principal 430
 - , rotation 32, 39, 41, 43, 48, 50, 63, 68
 - , Earth 85
 - , screw 47, 52
-
- ## B
- back
 - , EMF (electromotive force) 252, 260
 - , end 170
 - , projection 497
 - , error 497, 498
 - bag of words 515
 - balancing, white 306, 308
 - ballbot 112
 - barrel distortion 330
 - base
 - , force 269
 - , transform 199
 - Baxter robot 211, 277
 - Bayer
 - , filtering 294
 - , pattern 293

- Beer's law 289
 behavior-based robot 127
 Beidou (satellite navigation system) 153
 bi-quaternion (see *dual quaternion*)
 bias 88
 bicycle model 100, 107, 144, 145, 575
 bifilar pendulum 279
 bimodal distribution 416
 binarization 415
 binary
 - , classification 415
 - , image 371
 - , robust invariant scaleable keypoint (BRISK) 462
 - , segmentation 421
 blackbody 305
 - , radiator 288, 313
 black level 389
 blade flapping 115
 blend 72
 - , parabolic 72
 blob analysis 454
 body
 - , acceleration estimation 83
 - , moving 68
 body-fixed frame 39, 55, 70, 79, 115
 Boltzmann constant 288
 boundary 387, 419
 - , curvature 436
 - , detection 398
 - , effect 380
 - , gamut 299
 - , pixel 434
 bounding box 427
 Braitenberg vehicle 126
 breaking, stiction 252
 Bresenham algorithm 181
 BRISK (see *binary robust invariant scaleable keypoint*)
 Buffon's needle problem 174
 bug algorithm 128
 bundle adjustment 184, 355, 498–503, 521–523, 527
-
- C**
- C-space 56
 calibration
 - , Bouguet's 331
 - , camera 10, 319, 326, 331
 - , matrix 325, 333, 510
 - , nonlinear method 335
 - , sensor 88
 - , target 308
 camera 170
 - , affine 353, 503
 - , array 13, 349
 - , omnidirectional 326, 349
 - , panoramic 349
 - , baseline 483, 493
 - , calibration 10, 319, 326, 331
 - , homogeneous transform method 331
 - , matrix 325, 333, 510
 - , nonlinear method 335
 - , canonic 569
 - , catadioptric 340–343, 345, 355, 565, 570
 - , equiangular 341
 - , toolbox 341
 - , CCD 293, 294
 - , cellphone 324
 - , center 332, 350, 481
 - , central-perspective 323
 - , CMOS 285
 - , decomposition 334
 - , digital 293, 311
 - , DSLR (digital single-lens reflex) 364
 - , dynamic range 294, 365
 - , Euclidean 352
 - , finite projective 352
 - , fisheye lens 337, 339, 346
 - , frame 320, 321, 323
 - , global shutter 364
 - , high dynamic range 294
 - , hyperspectral 315
 - , image
 - , motion 542
 - , plane 321, 324
 - , infra-red 315, 508
 - , lens 321
 - , light-field 348, 350
 - , location determination problem 334
 - , LWIR (long-wavelength infra-red) 315
 - , matrix 323, 325–327, 331–333, 352, 469, 503, 527
 - , model 10
 - , modeling 319–344
 - , motion 454, 479, 481, 510, 521, 542, 547, 548, 552, 569
 - , multispectral 294
 - , nonperspective 352, 353
 - , orientation 327, 481
 - , panoramic 286, 308, 326, 348, 349
 - , parameter
 - , extrinsic 331, 333, 353, 503
 - , intrinsic 326, 331, 477, 480, 503
 - , perspective 319, 338, 340, 343, 344, 348, 350, 503, 565, 573
 - , pin-hole 319, 320, 349
 - , plenoptic 348
 - , pose 175, 326, 479, 521, 524, 538, 539, 541
 - , reflector-based 337, 340
 - , resectioning 354
 - , retreat 554, 565
 - , RGBD 509
 - , rolling shutter 364
 - , sensor 292, 313, 314
 - , SLR (single-lens reflex) 366
 - , spherical 342, 343, 570–572, 576, 578
 - , stereo 6, 483, 492, 496, 521
 - , thermographic 315
 - , time-of-flight 526
 - , ultraviolet 315
 - , unified model 344
 - , velocity 542–544, 547, 551, 552, 556–559, 567
 - , verged 471
 - , video 311
 - , wide-angle 286, 354, 546, 565
 Canny edge operator 384
 canonical image coordinate 322
 car 119–121
 Cardan angle sequence 36
 Cartesian
 - , coordinate system 22
 - , geometry 19
 - , motion 77, 211, 214, 238, 554

- , plane 19
- , point 179
- , trajectory 91, 214, 224
- catadioptric camera 340–343, 345, 354, 355, 565, 570
- cathode ray tube (CRT) 295
- caustic 341, 348
- CCD sensor 364
- celestial navigation 152
- CenSurE descriptor (see *center surround extremas*)
- census
 - , metric 391, 462
 - , transform 391, 489
- center
 - , of mass 64, 68, 115, 253, 264
 - , surround extremas (CenSurE) descriptor 462
- central
 - , imaging 340, 346, 348
 - , moments 429, 603
 - , perspective model 321
- centripetal
 - , acceleration 70
 - , force 264
- chamfer matching 401
- character recognition 418, 436
- characteristic scale 449
- charge well 364, 365
- Chasles theorem 52
- child region 432
- chi-squared (χ^2) distribution 160, 633
- Cholesky decomposition 590
- chroma keying 373
- chromatic aberration 330
- chromaticity 305, 312
 - , coordinate 298, 300
 - , D_{65} 306
 - , diagram 298, 300
 - , plane 299
 - , space 297, 298
- CIE (see *Commission Internationale de l'Eclairage*)
- circle 76, 606
 - , feature 544, 557
 - , of confusion 321
- circularity 434, 435, 454
- city block distance 130
- classification 415
 - , binary 415
 - , color 419
 - , grey-level 415, 416
 - , pixel 418, 421, 423
- cleaning up 491
- closed-form solution 205
- clothoid 101
- clustering
 - , algorithm 455
 - , k -means 421, 423, 514, 515
 - , of data 455
- CML (see *concurrent mapping and localization*)
- CMOS sensor 329, 364
- coarse-to-fine strategy 404
- coefficient
 - , Coulomb 272
 - , ellipse 557
 - , filter 376
 - , Fourier 436
 - , viscous friction 252, 272
- colatitude 342, 570
- collineation 608
- color 291
 - , blindness 295
 - , change 308, 363
 - , classification 419
 - , constancy 287, 307
 - , filter 293, 295
 - , gamut 299
 - , image 312, 361, 424, 460
 - , intensity 375, 382
 - , matching
 - , experiment 297
 - , function 297, 298, 300, 312, 316
 - , measuring 294
 - , name 300
 - , opponent 293
 - , plane 362, 373, 377, 448
 - , primary 294, 296
 - , reproduction 295, 297
 - , saturation 297, 301, 302
 - , segmentation 419
 - , space 301, 312
 - , HSV 301
 - , $L^*a^*b^*$ 303, 312
 - , L^*C^*h 301
 - , $L^*u^*v^*$ 303, 303
 - , opponent 303
 - , perceptually uniform 303
 - , XYZ 300, 301, 312
 - , $Y_C B_C R_C$ 303
 - , YUV 303
 - , spectral 298
 - , temperature 306, 314
- Color Checker 313
- colorimetry 298
- column space 591
- Commission Internationale de l'Eclairage (CIE) 298
 - , color space
 - , L^*C^*h 301
 - , $L^*u^*v^*$ 303
 - , standard primary colors 294, 298
 - , XYZ primary 300
- compass 41, 85, 108, 151, 153, 155, 164, 575
- compensation, gravity 118
- compound
 - , eye 285
 - , lens 321
- compression
 - , format 361, 363, 365
 - , gamma 311
 - , image 361, 363, 445
- computed torque control 274
- concurrent mapping and localization (CML) 167
- condition number (see *matrix condition number*)
- cone 351, 607
 - , cell 292, 293
 - , projection 351
- confidence test 164
- configuration
 - , change 216, 217
 - , kinematic 198, 208, 209, 215, 216, 238
 - , of a system 55
 - , space 55, 56, 114, 119, 121, 145, 198, 201, 210, 211
 - , zero-angle 197

- conic 322, 344, 352, 606
 - , projection 351
 - conjugate point 464, 467, 468, 470, 471, 475, 479, 483
 - connected component
 - , analysis 424, 425, 438
 - , graph 139, 642
 - , image 424, 426
 - connectivity 454
 - , analysis 424
 - consistency, left-right check 487
 - constant
 - , Boltzmann 288
 - , Planck 288
 - constraint
 - , epipolar 468, 484, 521, 522
 - , geometric 468
 - , nonholonomic 101, 111
 - , rolling 121
 - , smoothness 526
 - control
 - , feedback 262
 - , feedforward 118, 260, 262, 272, 273
 - , flexible transmission 13
 - , force 275
 - , independent joint 251
 - , integral
 - , action 259
 - , windup 280
 - , joint 251, 262
 - , loop, nested 251
 - , mobile robot 102–109
 - , model-based 192
 - , operational space 275, 276
 - , proportional 103, 104, 106, 257
 - , derivative 116–118
 - , integral 118, 260, 261
 - , resolved-rate motion 237, 248
 - , shared 7
 - , space 275, 276
 - , torque 272
 - , computed 274
 - , feedforward 273
 - , traded 7
 - , velocity 102, 257, 261
 - , vision-based (visual servo) 9, 11, 535
 - convolution 377, 383
 - , kernel 377, 382, 387, 393
 - , properties 377
 - coordinate
 - , frame 17, 18, 22
 - , 2-dimensional 19
 - , 3-dimensional 19
 - , end-effector 194
 - , global 181
 - , moving 68
 - , right-handed 31
 - , velocity 68
 - , generalized 55, 100, 109, 113, 119, 120, 194, 263
 - , homogeneous 604
 - , image 322
 - , plane 543
 - , joint 198, 218, 229, 263
 - , normalized 322, 543
 - , Plücker 52, 54, 350, 596
 - , point 22, 26, 47, 51
 - , random 641
 - , system 19
 - , vector 17–19, 587, 595, 604
 - Coriolis
 - , acceleration 70, 91
 - , force 263, 264, 267, 275
 - corner
 - , detector
 - , classical 443
 - , Harris 445, 449, 452, 456
 - , interest operator 443
 - , Noble 445
 - , Plessey 445
 - , scale-invariant 448
 - , scale-space 449
 - , Shi-Tomasi 445
 - , feature (see also *point feature*) 446, 448, 461, 521
 - , Harris 445, 448, 449, 452, 460–462, 520
 - , point 443, 446, 448, 461
 - , strength 445, 448
 - cornerness 445, 446
 - correlation 376, 377
 - , covariance 154, 632, 638
 - correspondence 461, 505, 506, 508, 521, 557, 559
 - , candidate 463, 472
 - , closest-point 507
 - , feature 460
 - , point 180, 471, 484, 522
 - , problem 459, 508, 556, 557
 - cost map 134
 - Coulomb friction 252, 253, 255
 - covariance
 - , correlation 154, 632, 638
 - , ellipse 160, 166, 633
 - , matrix 154, 156, 158, 160, 161, 163, 165, 167, 169, 170, 176, 632
 - , extending 165
 - crack code 434
 - cropping 401
 - CRT (see *cathode ray tube*)
 - curvature 141, 444, 448
 - , boundary 436
 - , principal 444
 - cybernetics 1, 4, 126, 147
-
- ## D
- D* 134
 - D_{65}
 - , chromaticity 306
 - , white 304, 305, 306, 312
 - d'Alembert force 69
 - damped inverse 240
 - data
 - , association 164, 460, 471, 472
 - , error 153, 164
 - , laser scan 179
 - , type 57, 58
 - dead reckoning 97, 151, 155
 - decimation, image 402
 - declination
 - , angle 85
 - , image 402
 - , magnetic 85
 - decoding, gamma 311

- decomposition 478
 - , camera 334
 - , Cholesky 590
 - , image 403
 - , matrix 525
 - , plane 478
 - , RQ 327
 - , spectral 591
 - , value 506
- Deep Phreatic Thermal Explorer (DEPTHX, AUV) 120, 121
- definition
 - , eigenvalue, eigenvector 41
 - , frame 70
 - , Mahalanobis distance 633
 - , robot 5, 126, 130
 - , white 306
- degree of freedom (DOF) 39, 56, 73, 114, 120, 121, 191, 193, 195, 208, 210, 231, 234, 236, 240–242, 542, 562, 565, 577
- Denavit-Hartenberg
 - , notation 196, 197, 217, 218, 221, 229
 - , modified 218
 - , parameter 197, 200, 227
- depth of field 321
- DEPTHX (see *Deep Phreatic Thermal Explorer*)
- derivative
 - , of Gaussian 384
 - , kernel 444
 - , orientation 64, 68, 118
 - , pose 63, 64
 - , quaternion 64
 - , time 63
- descriptor 462
 - , ACF (aggregate channel feature) 462
 - , BRISK (binary robust invariant scaleable keypoint) 462
 - , CenSurE (center surround extremas) 462
 - , FREAK (fast retina keypoint) 462
 - , Harris 461
 - , HOG (histogram of oriented gradients) 462
 - , MSER (maximally stable extremal region) 419, 438, 454, 462
 - , ORB (oriented FAST and rotated BRIEF) 462
 - , shape 433
 - , SIFT (scale-invariant feature transform) 462
 - , SURF (speeded up robust feature) 453, 462, 463, 472, 478, 479, 496, 514–516, 524, 556
 - , VLAD (vector of locally aggregated descriptors) 456
- detector 462
 - , AGAST 462
 - , corner (see also *corner detector*) 443, 445, 448, 449, 452, 456
 - , edge 384, 392, 407
 - , FAST 454, 462
 - , Harris 445, 447–449, 452, 456, 460, 461, 462, 520, 524, 527, 556
 - , Noble 445
 - , Shi-Tomasi 462
 - , SIFT (scale-invariant feature transform) 456, 462, 524
 - , SURF (speeded up robust feature) 452, 453, 456, 460, 462, 524, 252, 527
 - , zero crossing 387
- determinant 49, 235, 240, 445, 591
 - , of the Hessian 445
- dichromatic reflection 310
- difference of Gaussian 385
- differential, kinematics 229
- differentiation 384
- digital single-lens reflex (DSLR) camera 366
- Dijkstra method 132
- dimension 17
 - , curved 17
 - , intensity 301
 - , singleton 362
- diopter (see also *focal length*) 321
- Dirac function 313
- direction 611–613
- direct linear transform 354
- disparity 483, 487
 - , image 484, 487
 - , space image (DSI) 485, 489
- displacement
 - , rigid body 52, 53
 - , spatial 67, 245
- distance 164
 - , Euclidean 18, 130, 303, 312, 399, 400, 421, 423, 433, 461–463, 642
 - , Hamming 391
 - , Mahalanobis 164, 593, 633
 - , Manhattan 130, 587
 - , threshold 139, 464
 - , transform 130, 134, 135, 137, 399, 400
- distortion
 - , barrel 330
 - , correction 330
 - , decentering 330
 - , geometric 330
 - , hard iron 87
 - , keystone 509
 - , lens 330, 353, 405, 472, 496, 502
 - , map 336
 - , modeling 331
 - , perspective 391, 460, 509
 - , pincushion 330
 - , radial 330, 337
 - , rolling shutter 364
 - , soft iron 87
 - , shape 353, 509, 510
 - , tangential 330
 - , vector 406
- distribution
 - , bimodal 416
 - , chi-squared 633
 - , von Mises 156
- DOF (see *degree of freedom*)
- DoG kernel 384, 385
- DoH 445
- double cover 499
- down hill 623
- drag, aerodynamic 115
- DSI (see *disparity space image*)
- DSLR camera (see *digital single-lens reflex camera*)
- dual
 - , number 55
 - , quaternion 55
- Dubbins path 101
- dynamic range 365
- dynamics 251
 - , error 274
 - , forward 116, 118, 251, 271, 272
 - , inverse 263, 273, 274
 - , quadrotor 115, 116
 - , rigid-body 263, 272

-
- ## E
- Earth
 - , diameter 81
 - , gravity 82
 - , shape 81
 - , surface 70, 79, 512
 - east-north-up (ENU) 79
 - eccentricity 344, **600**
 - edge
 - , detection 377
 - , detector 392
 - , Canny 384, 407
 - , preserving filter 392
 - effect
 - , Eötvös 91
 - , jello 364
 - , picket fence 486
 - effective inertia 256
 - efficiency, quantum 364
 - EGNOS (satellite network) 153
 - eigenvalue 41, 160, 236, 270, 271, 430, 444, 503, **590**
 - eigenvector 41, 430, 503, **590**
 - EISPACK project 9
 - EKF (see *extended Kalman filter* and *Kalman filter*)
 - EKF SLAM (see *Kalman filter, extended, SLAM*)
 - elasticity, joint 276
 - ellipse 159, 321, 352, 537, 556, 557, **599**, 606, 633
 - , canonical 598, 599
 - , coefficient 557
 - , confidence 167, 168
 - , covariance **160**, 166, 633
 - , drawing 601
 - , equation 633
 - , equivalent 429–431
 - , error **160**, 163
 - , inertia of **603**
 - , parameter 557, 558
 - , rotated 633
 - , size 632
 - , velocity 235, 244
 - ellipsoid 351, **599**, 600, 607
 - , equation 633
 - , force 244, **245**
 - , hyper- 270
 - , shape 236
 - , surface 235, 245, 632
 - , velocity 244
 - , rotational 236
 - , volume 236, 601
 - , wrench 245
 - Elsie (robot) 95, 125
 - encoder 255, 256
 - encoding, gamma 306, 311, 312, 372
 - end-effector **193**
 - , coordinate frame 232
 - , force 244
 - , inertia 275
 - , torque 244
 - , velocity 229, 230
 - end-point
 - , closed-loop 537
 - , open-loop 537
 - ENU (see *east-north-up*)
 - Eötvös, effect 91
 - ephemeris 152
 - epipolar
 - , constraint 468, 484, 521, 522
 - , line **464–468**, 470, 471, 473, 479, 483, 525
 - , plane **464**, **465**
 - epipolar-aligned image **496**
 - epipole 466, 467
 - equal-energy white 306
 - equation
 - , differential 51
 - , ellipse 602, 604
 - , ellipsoid 548
 - , Eulers rotation **68**
 - , line 595, 605
 - , motion 101, 111, 271
 - , Euler 116, 263
 - , rigid-body 251, **263**
 - , optical flow 544, 570
 - , Planck radiation 288
 - , plane 504, 556
 - , solving system 621
 - , sparse nonlinear 501
 - , thin lens 321, 336
 - equiangular mirror **340**
 - equivalence principle 70
 - equivalent ellipsoid **503**
 - error 49, 50, 169, 170
 - , back projection **497**, **498**
 - , cumulative 170
 - , edge 172
 - , ellipse 159, 166, 168
 - , ICP (iterated closest point) 182
 - , position 251
 - , reprojection 502
 - , squared 501
 - , vector 628
 - essential matrix **468**, 470, 477, 480
 - estimation **154**
 - , camera
 - , motion 454
 - , pose 524
 - , Monte-Carlo 157, 175, 183
 - , pose 83, **334**, 536–538, 541, 556, 575
 - , RANSAC (random sampling and consensus) 471, 472, 476, 478, 504
 - , stereo 443
 - , SaM (structure and motion) 498, 578
 - ethics 7
 - Euclidean
 - , camera 352
 - , coordinate 29, 467, 468, 604
 - , distance **18**, 130, 303, 312, 399, 400, 421, 423, 433, 461–463, 642
 - , geometry 18, 19, 22, **595**
 - , group **21**, **27**, 46
 - , homography 477, 510, 511
 - , length **587**
 - , line 595
 - , plane 19, 605
 - , point 29, 595, 605, 606
 - , reprojection error 501
 - , space 19, 55, 595, 605, 608
 - , transformation **608**, **609**
 - Euler
 - , acceleration **70**

- , angle 36, 37, 38, 40, 59, 75, 196, 232, 233, 247, 499, 571
 - , singularity 39
 - , force 70
 - , motion equation 68, 116, 263
 - , rotation theorem 32, 33, 35–37, 613
 - EV (see *exposure value*)
 - EXIF file format 363, 510
 - explicit complementary filter 88, 89
 - exponential
 - , coordinate 43, 233, 481, 626
 - , rate 233
 - , mapping 50, 52
 - , matrix 25, 26, 43, 51
 - , product of 196, 200, 201
 - exposure 388, 461
 - , control 342
 - , interval 364
 - , time 321, 363, 364
 - , value (EV) 364, 365
 - extended Kalman filter (EKF, see also *Kalman filter*) 88, 90, 157, 169, 619, 638
 - exteroceptive sensor 5, 170
 - extromission theory 287
 - extrinsic parameter 503
 - eye 285, 287
 - , compound 285
 - , cone cell 292, 293
 - , dynamic range 365
 - , evolution 285
 - , fovea 293
 - , human 292
 - , lens-based 285
 - , reflector-based 285
 - , retina 293
 - , rod cell 365
 - , secondary 285
 - , sensitivity 301
 - , tristimulus 312
 - eye-in-hand 537
- ## F
- f*-number (inverse aperture diameter) 321, 364
 - FAST detector 454, 462
 - fast retina keypoint (FREAK) descriptor 462
 - FastSLAM (see also *SLAM* and *Rao-Blackwellized SLAM*) 169
 - feature
 - , blob 431
 - , circle 554, 557
 - , classification 415
 - , corner 446, 448, 461, 521
 - , correspondence 460
 - , depth 551
 - , description (see also *descriptor*) 445, 452, 453, 461
 - , detection (see also *detector*) 399, 449
 - , extraction 9, 286, 413
 - , Harris corner 445, 452
 - , image 332, 335, 413, 556
 - , line 413, 438, 446, 556
 - , map 163, 168
 - , moment 428
 - , point (see also *point feature*) 443, 449, 461
 - , region 413, 415
 - , scale 451
 - , scale-space 449, 452
 - , sensitivity matrix 542
 - , shape 435
 - , vector 432, 434
 - , vector-valued 415
 - feedback control 118, 260–262
 - feedforward control 118, 260, 262, 272, 273
 - fibre-optic gyroscope (FOG) 80
 - fictitious force 69, 83
 - field
 - , magnetic, intensity 86, 87
 - , of view 326, 327, 336
 - , robot 3, 96
 - file 172
 - , EXIF 363, 510
 - , image 360, 363
 - , raw 294
 - , JFIF 311
 - , JPEG 363
 - , MEX 584
 - , video 365
 - fill factor 329, 364
 - filter
 - , Bayer 293, 294
 - , coefficient 376
 - , complementary explicit 88, 89
 - , edge preserving 392
 - , Kalman 90, 91, 157, 162–164, 169, 175, 182, 184, 636
 - , extended (EKF) 88, 90, 157, 169, 619, 638
 - , unscented (UKF) 184
 - , Kalman-Bucy 637
 - , low-pass 384
 - , anti-aliasing 407
 - , spatial 403
 - , median 407
 - , particle 169, 175–178
 - , spatial 376
 - fish-eye lens
 - , camera 337, 339, 346
 - , projection model 338
 - flow
 - , current 85
 - , field 544
 - , optical 521, 544, 552, 553, 565, 570, 572
 - flux
 - , line, magnetic 85
 - , luminous 291, 294
 - , magnetic 85, 383
 - , visual 287
 - focal
 - , length 321, 331, 334, 364, 486
 - , point 320, 340, 341, 344, 348, 514
 - focus 319, 321, 330, 331
 - FOG (see *fibre-optic gyroscope*)
 - font, Hershey 220
 - force 52, 68, 244, 251
 - , apparent 69
 - , control 275
 - , Coriolis 263, 264, 267, 275
 - , d'Alembert 69
 - , ellipsoid 244, 245
 - , fictitious 69, 83
 - , gyroscopic 275
 - , inertial 69

- , pseudo 69
 - , translational 69
 - foreshortening 321, 509
 - form, homogeneous 29
 - formula
 - , Planck 288
 - , Rodrigues rotation 37, 42, 43, 52, 53, 61, 66, 613
 - forward
 - , dynamics 116, 271
 - , kinematics 193, 194, 201, 204, 230
 - , instantaneous 231
 - fovea 293
 - frame
 - , body-fixed 55, 70, 79
 - , coordinate 17, 18, 22
 - , key 503
 - , reference 69
 - , inertial 68, 69, 79, 83
 - , noninertial 70
 - , right-handed coordinate 31
 - , world coordinate 18, 79
 - FREAK (see *fast retina keypoint descriptor*)
 - Freeman chain code 434, 455
 - Fresnel reflection 310
 - friction 251–253, 262, 263, 268
 - , aerodynamic 115
 - , Coulomb 252, 253, 255, 268, 272
 - , stiction 252
 - , viscous 246, 252, 253, 255, 268, 271, 272
 - front end 170
 - fronto-parallel 321, 433, 510, 539, 541, 545, 549
 - frustum 326, 334
 - function
 - , Cauchy-Lorentz 504
 - , Dirac 313
 - , Gaussian 631
 - , Huber loss 625
 - , observation 164
 - , plenoptic 349
 - , probability density (PDF) 153, 160, 161, 175, 631
 - , Gaussian 175
 - , scalar 617
 - , signed distance 400
 - , Tukey biweight 625
 - fundamental matrix 466, 470, 525
 - fusion, sensor 87, 88, 163
- ## G
- gait pattern 225
 - Galileo (satellite navigation system) 153
 - gamma
 - , compression 311
 - , correction 310
 - , decoding 311, 362, 372, 373
 - , decompression 311
 - , encoding 311, 372, 407
 - , sRGB 311, 372
 - gantry robot 191
 - Gaussian
 - , distribution 635, 636
 - , function 378, 383, 631, 633
 - , width 378, 380
 - , kernel 386, 403, 444, 449, 451
 - , multivariate 632
 - , noise 157, 160, 164, 332, 335, 504, 507, 636, 637
 - , probability 160, 164, 633
 - , properties 380
 - , random variable 631, 636, 638
 - , smoothing 427
 - gearbox 254–256
 - generalized
 - , coordinate 55, 100, 109, 113, 119, 120, 194, 263
 - , joint 198, 218, 263
 - , forces 263
 - , joint 244, 246, 263, 264, 266, 268, 269
 - , matrix inverse 592
 - , Voronoi diagram 136, 399
 - generator matrix 612, 614
 - Genghis (robot) 147
 - geomagnet 85
 - geometric
 - , distortion 330
 - , invariant 609
 - , Jacobian 231
 - , transformation 608, 609
 - geometry
 - , algebraic 50
 - , analytic 19
 - , Cartesian 19
 - , Euclidean 18, 19, 22, 595
 - Gestalt principle 426
 - gimbal 205
 - , lock 38, 208, 215, 234
 - , low-friction 80
 - Global Hawk unmanned aerial vehicle (UAV) 4, 114
 - Global Positioning System (GPS) 5, 6, 117, 151, 153, 165
 - , differential 153
 - , multi-pathing 153
 - , RTK 153
 - , selective availability 153
 - global shutter camera 364
 - GLONASS (satellite navigation system) 153
 - goal seeking 128
 - Google Maps™ 367
 - G protein-coupled receptor (GPCR) 292
 - GPS (see *Global Positioning System*)
 - gradient 382, 383, 462
 - , calculation 377
 - , descent 623, 624
 - , edge 382, 384, 385
 - , image 384, 443, 444, 459, 559
 - , intensity 417, 438
 - , squared 460
 - graph 136, 139, 426, 499, 641
 - , A* search 134, 139, 142, 643
 - , embedded 641
 - Grassmann's laws 297, 299
 - gravity 70, 84, 115, 251, 253
 - , compensation 118
 - , disturbance 260
 - , load 251, 260, 263–265, 271
 - , term 264
 - , torque 254, 264
 - , vector 84, 263
 - great circle 76
 - grey value 360, 361, 368–372
 - ground effect 115

group 504
 →, algebraic 612
 →, Euclidean 21
 →, Lie 611
 →, orthogonal 24, 34, 590
 gyroscope 38, 79, 87, 101, 155
 →, fibre-optic (FOG) 80
 →, ring-laser (RLG) 80
 →, strapdown 80
 →, triaxial 80

H

Hall effect 85
 →, sensor 85
 Hamming distance 391
 hard-iron distortion 87
 Harris
 →, corner feature 445, 448, 449, 452, 460–462, 520
 →, detector 445, 447–449, 452, 456, 460, 461, 462, 520, 524, 527, 556
 heading 85
 →, angle 87
 →, rate (see *yaw rate*)
 helicopter 121
 Hershey font 220
 Hessian 617, 618, 624
 →, approximate 618, 624
 →, determinant 445
 →, matrix 445, 502, 618
 histogram 361, 373, 416, 448, 462
 →, cumulative 176
 →, equalization 372
 →, image 369, 371
 →, normalization 372, 407
 →, of oriented gradients (HOG) 462
 hit and miss transform 398
 HOG (see *histogram of oriented gradients*)
 holonomic constraint 56
 homogeneous
 →, equation 622
 →, form 27, 466
 →, transformation 27, 46, 53, 54, 77, 199, 203, 324, 325, 328, 477, 481, 504, 605
 →, normalization 50, 539, 549
 →, SE(2) 27
 →, SE(3) 46
 homography 10, 164, 474–478, 496, 510, 512, 513
 →, Euclidean 477, 510, 511
 →, matrix 13, 474
 →, planar 474
 →, plane-induced 474
 →, projective 477, 510
 →, RANSAC (random sampling and consensus) estimation 478
 homothety 608
 Hough transform 440, 454, 556
 hovercraft 119–121
 HSV color space 301
 Huber loss function 625
 hue 297, 301, 302
 humanoid robot 3, 6
 hybrid
 →, trajectory 72
 →, visual servo 565
 hyperbola 606
 hyperboloid 351, 607
 hypersurface, quadric 607
 hysteresis threshold 385

I

IBVS (see *image-based visual servo*)
 ICP (see *iterated closest point*)
 ICR (see *instantaneous center of rotation*)
 ideal
 →, line 328, 605
 →, point 605, 606
 identity quaternion 45
 illuminance 294, 307
 illumination, infra-red 508
 image 367
 →, anaglyph 495
 →, binary 371
 →, compression 361, 363, 445
 →, coordinate, canonical 322
 →, decimation 402
 →, disparity 484, 485, 487, 489
 →, epipolar-aligned 496
 →, feature 413, 556
 →, extraction 369, 413
 →, file format 360
 →, gradient 444
 →, histogram 369
 →, Jacobian 542, 544, 551, 568, 570
 →, matching 514
 →, metadata 363, 486, 510
 →, moment 428, 506
 →, monochromatic 361
 →, noise 364, 407
 →, obtaining 359
 →, perspective 341, 372
 →, plane 321, 605
 →, discrete 324
 →, processing 12, 130, 136, 359, 579
 →, pyramid 403
 →, rectification 496
 →, region 424
 →, resizing 402
 →, retrieval 13, 454
 →, segmentation 415
 →, similarity 387, 443
 →, census 391
 →, nonparameteric 391
 →, rank transform 392
 →, sphere 342
 →, stabilization 514
 →, stitching 512
 →, subsampling 402
 →, warping 336, 345, 404–406, 496, 510, 513
 image-based visual servo (IBVS) 537, 538, 541
 →, polar coordinate 568
 →, spherical camera 570
 imaging
 →, catadioptric 340
 →, central 340, 346, 348
 →, perspective 321
 →, light field 350, 355
 →, low-light 350

- , noncentral 341
 - , nonperspective 13
 - , model 336
 - , panoramic 319
 - , perspective 321, 336, 337
 - , underwater 309
 - , unified 344, 345
 - , wide-angle 343, 354
 - impulse noise 392
 - IMU (see *inertial measurement unit*)
 - incandescence 287
 - inclination
 - , angle 85
 - , magnetic 85, 86
 - incremental replanning 134
 - inertia 253–255
 - , effective 256
 - , end-effector 275
 - , load 255
 - , matrix 116, 266, 503
 - , motor 255
 - inertial
 - , force 69
 - , measurement unit (IMU) 39, 87, 577
 - , navigation system (INS) 79, 87, 117
 - , reference frame 68, 69, 79, 83
 - , sensor 87
 - Inf 484
 - inflation, obstacle 132
 - infra-red
 - , camera 315, 508
 - , illumination 508
 - , near (NIR) 315
 - , radiation 287–289, 292
 - , short-wavelength (SWIR) 315
 - innovation 89, 162, 170, 637
 - INS (see *inertial navigation system*)
 - instantaneous center of rotation (ICR) 100, 109
 - integral
 - , dynamics 271
 - , windup 260
 - intelligence, artificial 14, 524
 - Intel RealSense R200 509
 - intensity 302
 - , change 392
 - , color 375, 382
 - , dimension 301
 - , edge 381, 387
 - , gamma encoded 311
 - , gradient 392
 - , illuminance 307
 - , light 125, 293
 - , linear wedge 311
 - , luminous 294
 - , magnetic field 85, 87
 - , ramp 367
 - , sinusoid 367
 - , surface 396
 - inter-reflection 310
 - interaction matrix 542
 - interest point 443
 - International Telecommunication Union (ITU) 298
 - interpolation 441
 - , linear 75
 - , orientation 75
 - , quaternion 60, 76
 - , rotational 76
 - , scalar 212
 - , unit-quaternion 76, 77
 - intrinsic parameter 468, 503
 - invariance 433, 453
 - , geometric 609
 - , property 454
 - , rotational 444, 462
 - , time 377
 - inverse
 - , aperture diameter (*f*-number) 321, 364
 - , dynamic control 274
 - , dynamics 263, 273, 274
 - , left-generalized 621
 - , pseudo 240, 242, 548, 549, 592, 621
 - iris 321
 - ISO camera setting 364
 - iterated closest point (ICP) 179, 182, 183, 505, 506, 521, 526
 - ITU (see *International Telecommunication Union*)
- ## J
- Jacobian, Jacobian matrix 215, 218, 229, 230, 247, 617, 619
 - , analytical 232, 233
 - , condition 234
 - , damped inverse 240
 - , ellipse feature 558
 - , end-effector coordinate frame 232
 - , feature 568
 - , geometric 231
 - , image 542, 544, 549
 - , feature 568, 570
 - , insertion 165, 167
 - , line feature 556, 557
 - , manipulability 234, 235
 - , manipulator 229, 231, 247, 263
 - , matrix 158, 172, 192, 215, 229, 230
 - , numerical approximation 619
 - , over-actuated robot 242
 - , point feature 548, 559, 568
 - , singularity 234, 240
 - , transpose 229, 245, 246
 - , under-actuated robot 241
 - , visual 10, 545
 - jello effect 364
 - jerk 70
 - JFIF file format 311
 - Johns Hopkins Beast (robot) 147
 - joint
 - , actuator 252
 - , angle 5, 13, 198
 - , control, independent 251
 - , elasticity 276
 - , position 275
 - , prismatic 193, 195
 - , revolute 193
 - , sliding 193
 - , space 198, 244
 - , trajectory 212
 - , velocity 229, 230
 - Joseph form 637
 - JPEG file format 363

K

- k*-means 514, 515
 - , algorithm 420
 - , clustering 421, 423, 514, 515
 - Kalman filter 90, 91, 157, 162–164, 169, 175, 182, 184, 636
 - , extended (EKF) 88, 90, 157, 169, 619, 638
 - , SLAM (EKF SLAM) 169
 - , gain 637
 - , unscented (UKF) 184
 - kd-tree 464, 506
 - kernel 376
 - , circular 423
 - , convolution 377, 382, 387, 393
 - , density approach 183
 - , Gaussian 386, 403, 444, 449, 451
 - , Laplacian 386, 449
 - , Laplacian of Gaussian (LoG) 385, 386, 452
 - , Mexican hat 387
 - , smoothing 378, 448
 - , Sobel 382–384, 407
 - key frame 503
 - keypoint 443
 - keystone 509
 - , distortion 509
 - kidnapped robot 178
 - Kinect sensor 508
 - kinematic
 - , configuration 198, 208, 209, 215, 216, 238
 - , model 101, 107, 111, 114, 143, 145, 202
 - kinematics 193
 - , differential 229
 - , forward 193, 194, 201, 204, 230
 - , instantaneous 231
 - , symbolic 206, 230
 - , inverse
 - , closed form 205
 - , numerical 206, 209, 245
 - , velocity 229
 - Klein quadric 607
-
- L**
 - $L^*a^*b^*$ color space 303, 312
 - $L^*u^*v^*$ color space 303
 - Lambertian reflection 309, 337
 - landmark 152, 164, 169, 182, 462, 499
 - , identity 164
 - , navigation 151
 - , observation 161
 - , point 497, 500
 - Laplacian of Gaussian (LoG) 385, 449, 451
 - , kernel 385, 386, 452
 - , response 449
 - laser
 - , odometry 179
 - , rangefinder 178, 179, 181
 - , noise 180
 - , scanner 170
 - lateral motion 100
 - lattice planner 140
 - latus rectum 344
 - law
 - , Beer 289, 309
 - , Grassmann's 297, 299
 - , lens 321, 336
 - , Newton
 - , first 69
 - , second 68, 70, 82, 115, 263, 279
 - , of robotics 1
 - , power 311
 - , Stefan-Boltzman 288, 317
 - , Wien displacement 288
 - LCD (see *liquid crystal display*)
 - least squares problem 240, 241, 246, 332, 472, 553, 621
 - , nonlinear 171, 501, 618, 624, 625
 - , rotation matrix 622
 - left-right consistency check 487
 - length focal 321, 364
 - lens 320
 - , anamorphic 366
 - , aperture 321, 331
 - , compound 321
 - , distortion 330, 353, 405, 472, 496, 502
 - , entrance pupil 332
 - , equation 321
 - , *f*-number 321, 364
 - , fisheye 337
 - , focal length 321
 - , iris 321
 - , law 321, 336
 - , shape 570
 - , simple 321
 - , telecentric 353
 - , thin 321
 - lens-based eye 285
 - lenslet array 351
 - Levenberg-Marquardt
 - , algorithm 246, 624, 625
 - , optimization 246, 627
 - lever arm effect 253
 - Lie
 - , algebra 53, 54, 611–614
 - , group 25, 50, 611, 611–614
 - light
 - , absorption 290, 308
 - , field camera 348, 350
 - , intensity 125, 293
 - , monochromatic 287
 - , solar spectrum 289
 - , structured 507
 - , visible 287
 - line 606
 - , 2D 595
 - , 3D 596
 - , epipolar 464–468, 470, 471, 473, 479, 483, 525
 - , equation 595, 605
 - , Euclidean 595
 - , feature 413, 438, 446, 556
 - , fronto-parallel 321
 - , ideal 328, 605
 - , of no motion 100
 - , Plücker 351, 596–598
 - , projection 329, 351, 607
 - linear segment with parabolic blend (LSPB) trajectory 72, 261, 262
 - linearization 617
 - , general 617
 - link 252
 - , effect 253

- , elasticity 276
- , mass 253, 264
- LINPACK project 9
- liquid crystal display (LCD) 295
- load 277
 - , gravity 251, 260, 263–265, 271
 - , inertia 255
- localization 9, 151, 167, 181, 520
 - , algorithm 520
 - , CML (concurrent mapping and localization) 167
 - , error 153
 - , laser-based 182
 - , Monte-Carlo 175
 - , problem 153, 154
 - , SLAM (simultaneous localization and mapping) 167, 169–171, 175
- locus, spectral 298–301
- LoG kernel (see *Laplacian of Gaussian kernel*)
- longitude problem 152
- longitudinal motion 100
- long-wavelength infra-red (LWIR) 315
- LORAN (radio-based localization system) 153
- LORD MicroStrain 79
- LSPB (see *linear segment with parabolic blend*)
- lumen 291
- luminance 297, 299, 301, 306, 310, 349
- luminance 290, 294
- luminosity 291
- luminous
 - , flux 291, 294
 - , intensity 294
- LWIR (see *long-wavelength infra-red*)

M

- machine vision 6
- Machine Vision Toolbox (MVTB) 9
- magnetic
 - , declination 85
 - , field 86, 87
 - , flux 85, 383
 - , inclination 85, 86
 - , north 85, 87
 - , pole 85, 86
- magnetometer 85, 87
- Mahalanobis distance 164, 593, 633
- Manhattan distance 130, 587
- manifold 611–613
- manipulability 215, 234–237, 548
 - , dynamic 269, 271
- manipulator (see *also robot*) 191
 - , Jacobian 231, 244, 263
 - , kinematics 229
 - , over-actuated 56, 240, 242
 - , serial-link, dynamics 251
 - , under-actuated 56, 210, 240, 241
- manoeuvre 120, 121
- manufacturing robot 3
- map 164, 169, 367
 - , building, laser-based 181
 - , distortiom 336
 - , feature 163, 168
 - , obstacle 131
 - , road 367
 - , using 160
- mapping 167
 - , CML (concurrent mapping and localization) 167
 - , exponential 50, 52
 - , point 56
 - , PTAM (parallel tracking and mapping) 175
 - , SLAM (simultaneous localization and mapping) 167, 169–171, 175
- Markov random field (MRF) algorithm 526
- Marr-Hildreth operator 387
- Mars rover 4, 6, 7, 527, 528
- mass 68, 277
 - , center of 64, 68, 115, 253, 264
 - , distribution 68
 - , link 253, 264
 - , payload 268
 - , proof 82
- matching
 - , function, color 297, 298, 300, 312, 316
 - , image 514
 - , stereo 485, 486, 491, 497
 - , trichromatic 296
- mathematical morphology 136, 393
 - , closing 396, 423
 - , dilation 394
 - , erosion 394
 - , hit and miss 398
 - , end point 399
 - , skeleton 399
 - , triple point 399
 - , opening 395, 421
- MATLAB®
 - , code 10
 - , command prompt 10
 - , matrix xxix
 - , MEX-file 584
 - , object 9
 - , software 9
 - , Toolbox 354, 355
 - , conventions xxix
- matrix 325, 588
 - , adjoint 65, 69, 201, 247, 597, 615
 - , adjugate 589, 607
 - , angular velocity 66
 - , anti-symmetric 589
 - , camera 323, 325–327, 331–333, 352, 469, 503, 527
 - , condition number 235, 548, 550, 593
 - , covariance 154, 156, 158, 160, 161, 163, 165, 167, 169, 170, 176, 632
 - , diagonal 161
 - , extending 165
 - , odometry 160
 - , sensor 161
 - , decomposition 525
 - , definite
 - , negative 618
 - , positive 618, 626
 - , diagonalization 591
 - , essential 468–470, 477, 480, 498, 522
 - , estimation 10, 471
 - , exponential 25, 26, 43, 51
 - , exponentiation 50
 - , feature sensitivity 542
 - , generator 612, 614
 - , Hessian 445, 502, 618
 - , homography 13, 474

- , identity 66
 - , indefinite 618
 - , inertia 116, 266, 503
 - , interaction 542
 - , inverse
 - , damped 240
 - , pseudo 240–242, 548, 549
 - , Jacobian 172, 229, 230
 - , logarithm 25
 - , MATLAB® xxix
 - , normalization 49
 - , orthogonal 24
 - , orthonormal 34, 49
 - , projection 323
 - , rank 234, 332, 467, 468, 546, 592
 - , rotation 24, 35, 42, 50, 66
 - , determinant 49
 - , normalization 67
 - , product 25
 - , singular value decomposition 506, 592, 622
 - , skew-symmetric 25, 42, 43, 37, 43, 50, 51, 63, 66, 589, 613
 - , augmented 614
 - , sparse 628
 - , transformation, homogeneous 52, 64
- MAV (see *micro air vehicle*)
- maximally stable extremal region (MSER) algorithm, descriptor 419, 438, 454, 462
- maximum
 - , torque 259
 - , velocity 72
- measurement
 - , odometry 156
 - , random 156
 - , strapdown inertial 87
 - , unit, inertial (IMU) 40, 87, 577
- mecanum wheel 112
- median filter 392
- MEMS (see *micro-electro-mechanical system*)
- metamer 294
- method
 - , Newton's 624
 - , Newton-Raphson 623
 - , roadmap 136
- MEX-file 584
- Mexican hat kernel 387
- micro-electro-mechanical system (MEMS) 80
- micro air vehicle (MAV) 114
- microlens array 350
- Mikrokopter (software project) 122
- minimization, nonlinear 623
- minimum-norm solution 210, 215, 242
- Minkowski
 - , addition 395
 - , subtraction 395
- mirror 340
 - , concave 337
 - , conical 341
 - , equiangular 340, 341
 - , shape 340, 570
 - , spherical 341
- missing parts problem 486
- mixed pixel problem 391, 489
- mobile robot 3, 95, 99, 573
- mobility 121
- model
 - , 3D 13
 - , bicycle 100, 107, 144, 145, 575
 - , camera 10
 - , geometric 13
 - , imaging 321
 - , central perspective 321, 344
 - , unified 344, 345, 347, 565
 - , kinematic 101, 107, 111, 114, 143, 145, 202
 - , motion 99, 109, 112, 114, 115, 140, 144, 155, 271, 635, 636
 - , nonlinear 88
 - , process 635
 - , quadrotor 115
 - , reflection, dichromatic 310
 - , screw 48
 - , unicycle 107, 111
 - , vehicle 107
- model-based control 272
- moment 52, 602
 - , feature 428
 - , image 428, 506
 - , central 429, 431, 434, 506
 - , invariant 433, 434, 455
 - , line 596
 - , matrix 506, 622
 - , normalized 434
 - , second 444
 - , of inertia 68, 264, 429, 603
 - , principal 430
 - , torque 68, 115, 116, 244, 269
 - , vector 30, 47, 52, 351, 596
- momentum, angular 68, 79
- monochromatic
 - , image 361
 - , light 287
- Monte-Carlo
 - , estimation 157, 175, 183
 - , localization 175
- MOOC (see *open online course*)
- Moore-Penrose pseudo inverse 592
- Moravec interest operator 443
- morphology (see *mathematical morphology*)
- mosaicing 512
- motion 63, 84
 - , axis of 73
 - , camera 479, 481, 510, 521, 542, 547, 548, 552, 569
 - , Cartesian 77, 211, 214, 238, 554
 - , complex 12
 - , control, resolved-rate 234, 238, 239
 - , discontinuity 78
 - , end-effector 238
 - , equation 68, 101, 111, 116, 251, 263, 271
 - , inertial frame 84
 - , joint-space 211, 216
 - , lateral 112
 - , longitudinal 100
 - , model 99, 109, 112, 114, 115, 140, 144, 155, 271, 635, 636
 - , multi-dimensional 73
 - , null-space 13
 - , omnidirectional 99, 112, 128, 140
 - , perceptibility 548
 - , planner 105
 - , resolved-rate 13
 - , rigid-body 27, 46, 47, 54, 611, 612
 - , incremental 67

- , rotational 51, 52, 68
- , screw 47, 48
- , segment 74
- , sickness 83
- , singularity 215
- , straight-line 214, 560
- , translational 30, 31, 51, 53, 68

motor 255, 256, 277

- , DC 251
- , high-torque 254
- , inertia 255
- , limit 259
- , servo 251
- , stepper 251
- , torque 252

MRF (see *Markov random field*)

MSER (see *maximally stable extremal region*)

multi-camera array 348

multi-pathing 153

multi-segment trajectory 74

MVTB (see *Machine Vision Toolbox*)

N

NaN 484, 492

nautical

- , angle 38
- , chronometer 152
- , mile 151

navigation 97, 122, 125, 419, 455

- , aerospace 44
- , algorithm 131
- , Beidou (satellite navigation system) 153
- , chart 153
- , dead reckoning 151
- , Galileo (satellite navigation system) 153
- , GLONASS (satellite navigation system) 153
- , GPS (Global Positioning System) 5, 6, 117, 151, 153, 165
- , inertial 63, 66, 79, 87, 117
- , landmark 151
- , map-based 125
- , marine 167
- , planetary rover 525
- , principles 151
- , radio 79
- , reactive 125, 126
- , satellite 5, 6, 117, 151, 153, 165
- , spacecraft 38, 80
- , system 79, 87, 117

Navlab project 122

NCC similarity measure 388, 389, 410, 526, 531

near infra-red (NIR) 315

NED (see *north-east-down*)

nested control loop 251

Newton's

- , first law 69
- , method 624
- , second law 68, 70, 82, 115, 263, 279

Newton-Euler method 263, 278, 279

Newton-Raphson method 623

Newtonian telescope 337

Niblack threshold 418, 454

NIR (see *near infra-red*)

Noble detector 445

node, graph 20, 139, 141, 144, 170, 480, 641

noise 88, 156, 180, 359, 383

- , Gaussian 157, 160, 164, 332, 335, 504, 507, 636, 637
- , image 364, 407, 472
 - , impulse 392, 407
 - , reduction 383, 396, 444
 - , salt and pepper 392
- , odometry 156, 158, 635
- , pixel 383, 397
 - , dark current 364
 - , nonuniformity 364
 - , shot 364
- , random 88, 156, 177
- , scanning laser rangefinder 180
- , sensitivity 386, 572
- , sensor 162, 175

noncentral imaging 341

nonholonomy, nonholonomic 99

- , constraint 101, 111
- , system 121

nonhomogeneous equation 619

nonlocal maxima suppression 384, 386, 393, 441, 445, 446, 648

nonparametric transform 489

normalization

- , histogram 369, 372, 407
- , homogeneous transformation 50, 539, 549
- , rotation matrix 49

normalized

- , image coordinate 322, 406, 468, 477, 543, 557, 569
- , moment 434

normal matrix 590

north

- , magnetic 85, 87
- , true 85

north-east-down (NED) 79

null space of matrix 242, 467, 546, 592, 622

number

- , denominate 17
- , dual 55
- , random 139, 174, 635

O

objective lens 321

observation 161

obstacle

- , inflation 130
- , map 131

occlusion 423

occupancy grid 128, 130, 131, 181

OCR (see *optical character recognition*)

odometer 155

odometry 155, 156, 170

- , differential 155
- , laser 179
- , noise 156, 158, 635
- , visual (VO) 13, 520–522
- , wheel 155

omnidirectional

- , camera 326, 349
- , motion 99, 112, 128, 140
- , vehicle 112
- , wheel 112

OmniSTAR satellite network 153

open online course (MOOC) 11, 12

- operational space 55
 - control 275, 276
 - operator 71
 - associative binary 21
 - asterisc 81
 - backslash 71, 558, 621, 623
 - binary arithmetic 372
 - Canny edge 384, 385
 - differential 384
 - edge 385
 - Gaussian 385, 452
 - group 612
 - Harris 462
 - interest 443, 456
 - inverse 67
 - Laplacian 384, 386
 - Marr-Hildreth 387
 - monadic 362, 372, 415
 - multiplication 54
 - Sobel edge 458
 - spatial 359, 393
 - displacement 67
 - linear 376
 - nonlinear 376
 - opponent color
 - space 303
 - theory 293
 - opsin 292, 293
 - optical
 - axis 40, 321, 325, 496, 509, 541, 554, 566, 568, 570
 - character recognition (OCR) 436
 - flow 521, 544, 552, 553, 565, 570, 572
 - derotation 553
 - optimization 173, 175, 182, 401, 526
 - algorithm 246
 - bundle adjustment 498
 - graph 175
 - Levenberg-Marquardt 246, 627
 - nonlinear 333, 354
 - pose graph 172–174, 183
 - problem 171, 206
 - ORB (see *oriented FAST and rotated BRIEF*)
 - orientation 17
 - 2-dimensional 23
 - 3-dimensional 32
 - camera 327, 481
 - derivative 64, 68, 118
 - end-effector 196
 - error 88
 - estimation 80, 84, 89
 - feature 462
 - interpolation 75
 - region 431
 - relative 506
 - vector 40
 - vehicle 101, 108, 575
 - oriented FAST and rotated BRIEF (ORB) feature descriptor 462
 - origin 17
 - orthogonal matrix 34, 589, 592
 - orthographic projection 353
 - orthonormal matrix (see *orthogonal matrix*)
 - orthophoto 514
 - Otsu threshold 417, 454
 - over-actuated robot 56, 240, 242
 - over-actuation 121, 240
-
- ## P
- panoramic camera 326
 - parabolic blend 72
 - paraboloid 351, 607
 - parallel
 - projection 353
 - tracking and mapping (PTAM) system 175
 - parallel-link robot 191
 - parameter
 - camera 325, 326, 331, 333, 353, 477, 480, 503
 - Denavit-Hartenberg 197, 200, 227
 - ellipse 557, 588
 - extrinsic 326, 503
 - intrinsic 326, 503
 - particle filter 169, 175
 - path 70, 131, 134, 367, 399
 - payload 13, 251, 262
 - effect 268
 - lift capability 115
 - mass 268
 - PBVS (see *position-based visual servoing*)
 - PDF (see *probability density function*)
 - peak 153
 - finding 369, 416
 - point 390, 489, 645
 - refinement 489
 - response 291–293, 295
 - velocity 72
 - pencil of lines 471
 - pendulum, bifilar 279
 - perceptibility, motion 548
 - perception 5, 285
 - perceptually uniform color space 303
 - perimeter 434
 - perspective
 - camera 319, 338, 340, 343, 344, 348, 350, 503, 565, 573
 - correction 13, 509
 - distortion 391, 460, 509
 - foreshortening 509
 - image 341, 372
 - synthetic 347
 - imaging 321, 336, 337
 - projection 319–322, 325, 328, 347, 353, 459, 466, 469, 542, 543
 - tracking 443
 - transformation 319
 - perspective-*n*-point (PnP) problem 334
 - photogrammetry 354, 524
 - photometric unit 291
 - photopic response 291
 - photopsin 292
 - photoreceptor 292
 - photosensor array 350
 - photosite 293, 324, 364, 365
 - phototaxis 126
 - picket fence effector 486
 - pin-hole camera 285, 320, 321
 - pincushion distortion 330
 - pitch
 - angle 37
 - screw 47, 52
 - pixel
 - array 350
 - boundary 434
 - classification 418, 421, 423

- , noise 364, 383, 397, 472
- , value, distribution 369
- planar
 - , homography 474
 - , robot 205
 - , surface 97, 119
 - , transformation 31, 609
- Planck
 - , constant 288
 - , radiation formula 288
- Planckian source 288
- plane 598, 607
 - , Cartesian 19
 - , chromaticity 299
 - , color 362, 373, 377, 448
 - , decomposition 478
 - , epipolar 464, 465
 - , equation 504, 556
 - , Euclidean 19, 605
 - , image 321, 324, 605
 - , principal 327
- planning
 - , algorithm 135
 - , map-based 130
 - , robot path 130, 134, 367, 399
 - , trajectory 147, 555
- plenoptic
 - , camera 348
 - , function 349
- Plessey corner detector 445
- Plücker
 - , coordinate 52, 54, 350, 596
 - , line 351, 596–598
- PnP (see *perspective-n-point*)
- point 17, 413
 - , 3D 31, 319
 - , Cartesian 179
 - , cloud 181, 184, 503, 504, 506
 - , 3-dimensional 181
 - , conjugate 464, 467, 468, 470, 475, 479
 - , coordinate 26
 - , homogeneous 51
 - , vector 22, 47
 - , corner 443, 446, 448, 461
 - , corresponding, correspondence 180, 471, 473, 474, 476, 478, 484, 487, 496, 507, 522
 - , detection 459
 - , edge 434, 439
 - , epipolar 473
 - , equation
 - , ellipsoid surface 235, 245
 - , line 605
 - , Euclidean 29, 595, 605, 606
 - , feature 443, 449, 461
 - , BRISK (binary robust invariant scaleable keypoint) 454
 - , extraction 10
 - , FAST 454, 462
 - , Harris 454, 520
 - , MSER (maximally stable extremal region) 419, 438, 454, 462
 - , scale-space 449, 452
 - , SIFT (scale-invariant feature transform) 452, 454, 462
 - , SURF (speeded up robust feature) 452, 454, 460, 462, 463, 472, 478, 479, 496, 514
 - , focal 320, 340, 341, 344, 348, 514
 - , homogeneous form 29
 - , ideal 605, 606
 - , image-plane 503
 - , instantaneous center of rotation (ICR) 100, 109
 - , interest 443, 444, 460
 - , iterative closest (ICP) 505
 - , landmark 497, 500
 - , line equation 605
 - , mapping 56
 - , moving to 102
 - , peak 390, 489, 645
 - , perimeter 435
 - , principal 325, 330, 331, 338, 340, 345, 347, 406, 480, 514, 543, 544, 549, 568
 - , salient 443
 - , set, matching 505
 - , spread function 321
 - , task space 56
 - , tool center (TCP) 203
 - , transformation 24
 - , triple 399
 - , vanishing 321
 - , vector xxix, 17, 22
 - , velocity, angular 64
 - , world 319, 322, 323, 325, 326, 331, 332, 459
- Poisson distribution 364
- polar-coordinate robot arm 196
- pole
 - , magnetic 85, 86
 - , rotational 30
- polynomial
 - , ellipse 600
 - , function of time 71
 - , matrix approximation 52
 - , trajectory 71
- pose 17, 55, 60, 170
 - , 2D 57
 - , 3D 58
 - , camera 175, 326, 479, 521, 538, 539, 541
 - , change 63
 - , derivative 63, 64
 - , end-effector 193, 229
 - , error 170, 245
 - , estimation 83, 334, 536–538, 541, 556, 575
 - , graph 170, 171
 - , optimization 172–174, 183
 - , SLAM (simultaneous localization and mapping) 167, 169–171, 175
 - , robot (see also *manipulator*) 179, 181
 - , singular 234
 - , trajectory 77
- position 17
- position-based visual servoing (PBVS) 537, 538
- positive definite 590
- posterior probability 157
- posterization 372
- power
 - , distribution, spectral (SPD) 317
 - , law 311
 - , series 52
- primary
 - , CIE (Commission Internationale de l’Eclairage) 294, 297, 300, 305
 - , color 294, 296
 - , standard 305

- PrimeSense camera 508
 - principal
 - , axis 430
 - , curvature 444
 - , moment 430
 - , plane 327
 - , point 325, 330, 331, 338, 340, 345, 347, 406, 480, 514, 543, 544, 549, 568
 - prior probability 157
 - probabilistic roadmap (PRM) 137
 - probability 11, 37, 154, 157, 174
 - , conditional 157
 - , density function (PDF) 153, 160, 161, 175, 631, 632
 - , Gaussian 160, 164, 633
 - , posterior 157
 - , prior 157
 - process noise 156, 636
 - Procrustes transform 609
 - product
 - , of exponential 200, 201
 - , of inertia 68, 429, 603
 - projection
 - , back 497, 498
 - , line 329, 351, 607
 - , matrix 323
 - , model 338
 - , orthographic 353
 - , parallel 353
 - , perspective 319, 321, 322, 328, 347, 353, 459, 466, 469, 542, 543
 - , weak 353
 - , point 320–324, 325, 327
 - , quadric 352, 607
 - , stereographic 345
 - projective
 - , homography 477, 510
 - , reconstruction 503
 - , transformation 321, 608
 - projector, speckle 509
 - Prometheus Project 122
 - proof mass 82
 - proprioception 546
 - proprioceptive sensor 5
 - pseudo
 - , force 69
 - , inverse 240, 242, 548, 549, 592, 621
 - , Moore-Penrose 592
 - , random numbers 174
 - PTAM (see *parallel tracking and mapping*)
 - Puma 560 robot 196, 202, 256, 276
 - pure
 - , pursuit 105
 - , quaternion 45, 55, 64
 - purple boundary 298
 - pyramidal decomposition 403
-
- Q**
- quadratic surface 351, 607
 - quadric 350, 351, 606, 607
 - , hypersurface 607
 - , Klein 607
 - , projection 607
 - quadrotor 56, 97, 99, 114, 120, 565, 576
 - , control system 117
 - , dynamics 115, 116
 - , model 115
 - quantum efficiency 364
 - quaternion 44
 - , computational efficiency 45
 - , conjugate 45
 - , convert to rotation matrix 45
 - , derivative 64
 - , double cover 44, 481
 - , dual 55
 - , identity 45
 - , interpolation 60, 76
 - , pure 45, 55, 64
 - , unit 44, 45, 47, 50, 55, 58, 64, 76, 499
 - quintic polynomial 71
 - quiver plot 384
-
- R**
- radial distortion 330
 - radiation
 - , absorption 289
 - , electro-magnetic 287
 - , infra-red 287–289, 292
 - , Planck formula 288
 - radiometric unit 291
 - radio navigation 79, 153
 - radius, turning 141
 - random
 - , coordinate 641
 - , dot pattern 508
 - , measurement 156
 - , noise 88, 156, 177
 - , number 139, 174, 635
 - , sampling 139, 145
 - , and consensus (RANSAC) 471, 472, 476, 478, 504
 - , variable 631
 - , Gaussian 631, 636, 638
 - rangefinder
 - , remission 179
 - , scanning laser 178, 179–181
 - rank
 - , filter 392
 - , matrix 234, 332, 467, 468, 546, 592
 - , transform 391, 392, 462, 489
 - RANSAC (see *random sampling and consensus*)
 - Rao-Blackwellized SLAM (see also *FastSLAM*) 169
 - rapidly-exploring random tree (RRT) 144, 145
 - rate
 - , angular 88
 - , exponential coordinate 233
 - , roll-pitch-yaw angle 76, 118, 233
 - , rotation matrix 64
 - ratio 268
 - , ambiguity 486
 - , aspect 413, 430, 431, 433
 - , gear 254, 264
 - raw image file 294
 - raxel 350, 481
 - recognition, character 418, 436
 - reconstruction 491
 - , affine 503
 - , projective 503
 - rectification 496

- recursive Newton-Euler 263
- redundant robot 56, 210, 226, 240
- Reeds-Shepp path 101
- reference
 - , frame 69
 - , inertial 68, 69, 79, 83
 - , noninertial 70
 - , system, attitude and heading (AHRS) 87
- reflectance, reflectivity 179, 180, 290, 307, 308, 608
 - , dichromatic 310, 316
 - , surface 290, 308, 310, 337
- reflection
 - , diffuse 309
 - , Fresnel 310
 - , geometric 609
 - , Lambertian 309, 337, 559
 - , model 310
 - , spectrum 290
 - , specular 180, 309, 337, 423, 424
- reflector-based
 - , camera 337, 340
 - , eye 285
- region
 - , area 428
 - , aspect ratio 430
 - , bounding box 427
 - , centroid 429
 - , child 435
 - , equivalent ellipse 429
 - , feature 413, 415
 - , image 424
 - , inertia matrix 429
 - , maximally stable extremal (MSER) 419, 438, 454, 462
 - , of interest 401
 - , orientation 431
- remission 179
- renormalization 55
- replanning, incremental 134
- representational singularity 233
- reprojection error 500
- resampling 176
- resectioning 152
- resizing 402
- resolved-rate motion control 234, 237
- response
 - , human eye 288, 289
 - , Laplacian of Gaussian 449
 - , peak 291–293, 295
 - , photopic 291
 - , position loop 262
 - , scotopic 291
 - , spectral 292–294, 296, 313, 315
 - , tristimulus 312
 - , velocity loop 258–260
- retinal
 - , molecule 292
 - , ganglion layer 293
 - , image plane coordinates 322
- retinex theory 307, 316
- RGBD camera 509
- rhodopsin 292
- right-hand rule 31
- rigid-body
 - , displacement 46, 52, 53
 - , dynamics 263, 272
 - , motion 27, 46, 47, 54, 67, 611, 612
- ring-laser gyroscope (RLG) 80
- roadmap 136
- robot (see also *manipulator*) 191
 - , arm 121
 - , model 200
 - , planar 194, 245
 - , polar-coordinate 196
 - , PUMA 195
 - , serial-link 196
 - , SCARA (Selective Compliance Assembly Robot Arm) 191, 195, 210
 - , Stanford 195
 - , Asimo humanoid 6
 - , base transform 203, 218
 - , Baxter 211, 277
 - , behavior-based 127
 - , definition of 5, 126, 130
 - , DEPTHX (Deep Phreatic Thermal Explorer, AUV) 120, 121
 - , Elsie 96
 - , end-effector 192
 - , field 3, 96
 - , gantry 191
 - , high-speed 276
 - , humanoid 3, 6
 - , joint
 - , modelling 255
 - , structure 195
 - , kidnapped 178
 - , law 1
 - , manipulability 215, 236
 - , manufacturing 3
 - , maximum payload 268
 - , mobile 3, 95, 99, 573
 - , over-actuated 56, 242
 - , path planning 131, 134, 367, 399
 - , parallel-link 191
 - , planar 205
 - , pose 179, 181
 - , Puma 560 196, 202, 256, 276
 - , redundant 56, 210, 226, 240
 - , Shakey 95
 - , service 3
 - , singularity 208, 215
 - , tele- 6
 - , tool transform 203, 204, 218, 222
 - , tortoise 95
 - , trajectory 169
 - , under-actuated 56, 210, 240, 241
 - , walking 221
 - , wrist 196, 215
- Rodrigues
 - , rotation formula 37, 42, 52, 53, 61, 66, 613
 - , vector 42
- roll angle 37
- roll-pitch-yaw angle 37, 38, 40, 232, 233
 - , rate 76, 118, 233
 - , singularity 38
 - , XYZ 37, 38, 214, 232
 - , YXZ 481
 - , ZYX 37
- rolling, constraint 121

- root, finding 622
 - Rossum's Universal Robots (RUR) 3
 - rotation, rotational 47, 50, 54, 608
 - , angle 25, 26, 31, 35, 37, 39, 43
 - , axis 32, 39, 41, 43, 48, 50, 63, 68
 - , direction 76
 - , formula 37, 42, 52, 53, 61, 66, 613
 - , incremental 66
 - , inertia 68
 - , interpolation 76
 - , invariance 444, 462
 - , matrix 24, 35, 36, 40, 42, 45, 50, 66, 232, 405, 511, 576
 - , determinant 49
 - , estimating 622
 - , least squares problem 622
 - , normalization 67
 - , product 25
 - , reading 35
 - , motion 51, 52, 68
 - , pole 30
 - , rate 64
 - , theorem, Euler's 32, 33, 35–37, 613
 - , torque 69
 - , twist 30
 - , vector 30
 - , velocity 63, 65, 69
 - row space 591
 - RQ decomposition 327
 - RRT (see *rapidly-exploring random tree*)
 - RTK GPS (see *Global Positioning System (GPS), RTK*)
 - rule, right-hand 31
 - RUR (see *Rossum's Universal Robots*)
- ## S
- saccule 83
 - SAD similarity measure 389, 392
 - salient point 443
 - salt and pepper noise 392
 - SaM (see *structure and motion*)
 - sampling
 - , artifact 402
 - , importance 176
 - , probabilistic 147
 - , random 139, 145
 - , Shannon-Nyquist theorem 402
 - , spatial 402
 - satellite
 - , navigation
 - , system 5, 6, 117, 151, 153, 165
 - , network 153
 - , view 367
 - saturation
 - , actuator 118
 - , color 297, 301, 302
 - , function 375
 - scalar 17, 54
 - , field 618
 - , function 617, 618
 - , interpolation 212
 - , multiplication 587
 - scale 384
 - , characteristic 449
 - , factor 88
 - , feature 451
 - , space 384, 403, 462
 - , spatial 384
 - scale-invariant feature transform (SIFT)
 - , descriptor 462
 - , detector 456, 462, 524
 - scaling 608
 - scanning laser rangefinder 178, 179, 181
 - , noise 180
 - SCARA (see *Selective Compliance Assembly Robot Arm*)
 - scene luminance 364
 - Schur complement 628
 - scotopic response 291
 - screw 47, 52
 - , axis 47, 52
 - , model 48
 - , motion 47, 48
 - , pitch 47, 52
 - , theory 52
 - SE(2) 27, 34
 - se(3) 53, 54, 614
 - SE(3) 46, 48, 53, 54, 73, 77, 479, 614, 615, 626
 - SEA (see *series-elastic actuator*)
 - segmentation 13, 396
 - , binary 421
 - , color 419
 - , graph-based 426
 - , image 415
 - , shape 528
 - selective availability 153
 - Selective Compliance Assembly Robot Arm (SCARA) 191, 195, 210
 - semi-global matching (SGM) 526
 - sensor 170
 - , acceleration 83, 87
 - , bias 88
 - , calibration 88
 - , camera 292, 313, 314
 - , CCD 364
 - , CMOS 364
 - , drift 88
 - , error 170
 - , fusion 88, 163
 - , Hall effect 85
 - , inertial 87
 - , Kinect 508
 - , noise 162, 175
 - , range and bearing 161
 - serial-link manipulator 193
 - series-elastic actuator (SEA) 276, 277
 - servo-mechanism 537
 - servoing
 - , visual 537, 572
 - , advanced 565
 - , image-based 536, 538, 541
 - , photometric 559
 - , position-based 536, 538
 - SfM (see *structure from motion*)
 - SGM (see *semi-global matching*)
 - shadow 314
 - , removal 313
 - Shakey (robot) 95
 - shape 322, 413, 423, 433
 - , change 13, 235, 245, 359, 401
 - , descriptor 433
 - , distortion 353, 509, 510

- , Earth 81
- , ellipse 556
- , ellipsoid 236
- , feature 435
- , filter 394
- , fitting 456
- , from moment 433
- , from perimeter 434
- , lens 570
- , mirror 340, 570
- , object 319, 393, 435
- , perimeter 454
- , segmentation 528
- , structuring element 394
- shared control 7
- shear, transformation 608
- Shi-Tomasi detector 462
- shift invariance 377
- short-wavelength infra-red (SWIR) 315
- SIFT (see *scale-invariant feature transform*)
- signed distance function 400
- similarity transform, transformation 591, 609
- similar matrix 591
- Simulink 11, 272
 - , block 101
 - , library 111
 - , diagram 536
 - , kinematics 214
- simultaneous localization and mapping (SLAM) 167
 - , back end 170, 174, 175
 - , EKF (extended Kalman filter) 169
 - , Fast 169
 - , front end 170, 174
 - , pose graph 167, 169–171, 175
 - , Rao-Blackwellized 169
 - , system, vision-based 175
- single-lens reflex (SLR) camera 366
- singleton dimension 362
- singular
 - , pose 234
 - , value 592
 - , decomposition 592
 - , vector 592
- singularity 37, 38, 208, 215
 - , angle
 - , Euler 39
 - , roll-pitch-yaw 38
 - , Jacobian 234, 240
 - , motion 215
 - , representational 233
 - , three angle representation 38
 - , wrist 208, 215
- singular value decomposition (SVD) 592, 621, 622
- skeleton 137, 203
 - , topological 136
- skeletonization 136, 137
- skew-symmetric matrix 25, 26, 27, 42, 43, 50, 51, 63, 66, 90, 351, 589, 606, 607, 613
 - , augmented 614
- skid steering 111
- SLAM (see *simultaneous localization and mapping*)
- SLR camera (see *single-lens reflex camera*)
- smoothing 377, 384
- smoothness constraint 526
- SO(2) 24, 611, 612
- so(3) 54, 233, 613
- SO(3) 34, 68, 73, 75, 81, 612, 613
- Sobel kernel 382
- soft-iron distortion 87
- solar spectrum 289
- solid angle 294, 326
- solution
 - , closed-form 205
 - , minimum-norm 242
 - , numerical 206
- solving system 621
- SOS (see *standard output sensitivity*)
- source, Planckian 288
- space
 - , affine 608
 - , chromaticity 297, 298
 - , color (see also *color space*) 301, 312
 - , configuration 55, 56, 114, 119, 121, 145, 198, 201, 210, 211
 - , control 275, 276
 - , Euclidean 19, 55, 595, 605, 608
 - , inertial reference equipment (SPIRE) 79
 - , joint 198, 212, 244
 - , operational 55
 - , control 275, 276
 - , resectioning 354
 - , scale 384, 403, 462
 - , task 55, 56, 210, 211
 - , vector 587
- sparse
 - , matrix 628
 - , stereo 479, 483, 492, 524, 552
- spatial
 - , aliasing 402, 486, 488
 - , displacement 67, 245
 - , filter 376
 - , operator 67, 359, 376, 393
 - , sampling rate 402
 - , scale 384
 - , velocity 64, 65, 69, 231, 232, 239, 542, 546, 573
 - , vector 64
- SPD (see *spectral power distribution*)
- special
 - , Euclidean group 21, 27, 46
 - , orthogonal group 24, 34, 590
- speckle projector 509
- spectral
 - , color 298
 - , decomposition 591
 - , locus 298–300
 - , power distribution (SPD) 317
 - , response 292–294, 296, 313, 315
- spectrum
 - , absorption 289, 290, 309
 - , D_{65} standard white 312
 - , illumination 307
 - , infra-red 292
 - , luminance 290, 294, 297, 312
 - , reflection 290
 - , solar 289
 - , visible 289
- specular reflection 180, 309, 337, 423, 424
- speculum, metal 337
- speeded up robust feature (SURF)
 - , descriptor 453, 462, 463, 472, 478, 479, 496, 514–516, 524, 556
 - , detector 452, 453, 456, 460, 462, 524, 252, 527

- spherical
 - , aberration 330
 - , camera 342, 343, 570–572, 576, 578
 - , image-based visual servo (IBVS) 570
 - , linear interpolation 76
 - , mirror 341
 - , wrist 199, 205, 207
 - SPIRE (see *space inertial reference equipment*)
 - spring 82, 277
 - , torsional 277
 - SSD similarity measure 389, 443, 514
 - stabilization, image 514
 - standard output sensitivity (SOS) 364
 - Stanford, robot arm 195
 - STAR (see *center surround extremas (CenSurE) descriptor*)
 - steering
 - , Ackermann 101, 123
 - , angle 101, 102, 141, 145
 - , mechanism 99
 - , skid 111
 - Stefan-Boltzman law 288, 317
 - steradian 326
 - stereo
 - , baseline 524
 - , camera 6, 483, 492, 496, 521
 - , estimation 443
 - , failure mode 485
 - , glasses 35
 - , matching 485, 486, 491, 497
 - , movie 495
 - , pair 483, 493, 495, 496
 - , perception 495
 - , sparse 479, 483, 492, 524, 552
 - , system 492, 507
 - , technique 552
 - , triangulation 522
 - , vision 479, 488, 491, 503, 507, 509, 524
 - stereographic projection 345
 - stereopsis 483
 - stiction 252
 - stop word 516
 - straight-line motion 214
 - strapdown
 - , configuration 80
 - , gyroscope 80
 - , inertial measurement 87
 - structure
 - , and motion (SaM) estimation 498, 578
 - , from motion (SfM) 498, 527
 - , tensor 444, 445, 448, 461
 - structured light 507
 - structuring element 393
 - subpixel interpolation 648
 - subsampling, image 402
 - subsumption architecture 127
 - subtraction, Minkowski 395
 - Sun spectrum 289
 - support region 451, 453, 462
 - suppression, nonlocal maxima 384, 386, 441, 445, 446
 - SURF (see *speeded up robust feature*)
 - surface 494
 - , 2D 319
 - , 3D 132
 - , Earth 70, 79, 512
 - , ellipsoid 235, 245, 632
 - , geometry 310
 - , hypersphere 235
 - , intensity 396
 - , luminance 290
 - , matte 310
 - , meshing 528
 - , planar 97, 119
 - , polished 180
 - , quadratic 351, 607
 - , reflectance 290, 308, 310
 - , reflective 337
 - , sphere 342, 344, 570
 - , textureless 509
 - , water 309
 - , writing on 220
 - SVD (see *singular value decomposition*)
 - Swedish wheel 112
 - SWIR (see *short-wavelength infra-red*)
 - symmetric matrix 266, 444, 589
 - system
 - , attitude and heading reference (AHRS) 87
 - , configuration 55
 - , coordinate 19
 - , homogeneous 622
 - , inertial navigation (INS) 79, 87, 117
 - , nonholonomic 121
 - , nonhomogeneous 621
 - , nonintegrable 121
 - , nonlinear 638
 - , under-actuated 120
 - , vestibular 80, 83, 546
-
- ## T
- tag, April 164
 - Tait-Bryan angle 38
 - tangential distortion 330
 - tangent space 612
 - task space 55, 56, 210, 211
 - taxis 126
 - Taylor series 444, 617
 - TCP (see *tool center point*)
 - telerobot 6
 - telecentric lens 353
 - temperature
 - , color 306, 314
 - , drift 88
 - template matching 484
 - tensor 587
 - , structure 444, 445, 448, 461
 - , trifocal 525
 - texture mapping 346, 494
 - theorem
 - , Chasles 52
 - , Euler's rotation 32
 - theory
 - , Lie group 25
 - , opponent color 293
 - , retinex 307, 316
 - , screw 52
 - , trichromatic 293
 - thin lens 321
 - thinning (also *skeletonization*) 136, 137

- threshold 376, 415, 418
 –, corner strength 448
 –, distance 139, 464
 –, local 418
 –, Otsu's method 417
 –, Niblack algorithm 418
 thresholding 371, 407, 415
 –, hysteresis 385
 thrust 115
 tie point 512
 time 63
 –, derivative 63
 –, exposure 321, 363, 364
 –, invariance 377
 –, of flight 508, 526
 –, series xxix
 –, varying pose 63, 70
 tone matching 513
 tool
 –, center point (TCP) 203
 –, transform 199, 203, 204, 218, 222
 toolbox
 –, functions 57–59
 –, obtaining 583
 top hat kernel 379
 topological skeleton 136
 topology, algebraic 50
 torque 251, 253, 254, 275
 –, control 272
 –, computed 272, 274
 –, feedforward 260, 272, 273
 –, disturbance 251
 –, end-effector 244
 –, gravity 254, 264
 –, maximum 259
 –, moment 68, 115, 116, 244, 269
 –, motor 252
 –, rotational 69
 trace of matrix 591
 traded control 7
 trajectory 70, 74, 76–78, 90, 139, 169, 209, 211, 223, 225, 251, 263
 –, Cartesian 91, 214, 224
 –, continuous 74, 220
 –, end-effector 251
 –, following 105, 140
 –, hybrid 72
 –, joint-space 212–214, 216
 –, lane-changing 102
 –, leg 221
 –, multi-axis 73
 –, multi-segment 74
 –, planning 147, 555
 –, polynomial 71
 –, pose 77
 –, robot 169
 transconductance 252
 transform
 –, base 199
 –, census 391, 489
 –, distance 130, 134, 135, 137, 399, 400
 –, nonparametric 391, 489
 –, planar 31
 –, Procrustes 609
 –, rank 391, 392, 462, 489
 –, SE(2) 31
 –, tool 199, 203, 204, 218, 222
 transformation
 –, affine 608
 –, conformal 322
 –, Euclidean 608, 609
 –, geometric 608, 609
 –, homogeneous 27, 46, 53, 54, 77, 199, 203, 324, 325, 328, 477, 481, 504, 605
 –, matrix 52, 64
 –, perspective 319
 –, planar 609
 –, point 24
 –, projective 321, 608
 –, SE(2) 27
 –, SE(3) 46
 –, similarity 608, 609
 –, wrench 244
 translation 46, 53, 54, 608
 transmission 251, 276, 309
 –, flexible 13
 –, mechanical 109
 transpose, Jacobian 246
 trapezoidal trajectory 72
 traversability 130, 134
 triangulation 152, 459, 497, 521
 triaxial
 –, accelerometer 83, 87
 –, gyroscope 80
 –, magnetometer 85
 trichromatic
 –, matching 296
 –, theory 293
 trifocal tensor 525
 triple point 136
 tristimulus 294–299, 301, 302, 304–306, 308, 311, 362
 –, eye 312
 –, response 312
 –, value 304, 315
 true north 85
 Tukey biweight function 625
 turning radius 100, 141
 twist 30, 48, 52, 53, 200, 247, 614
 –, axis 47
 –, Jacobian computing 247
 –, nonunit 31, 48
 –, rotational 30
 –, transforming 614
 –, unit 30, 48, 52, 54
 –, vector 30, 31, 47
 –, velocity 65, 247
-
- U**
 UAV (see *unmanned aerial vehicle*)
 UKF (see *unscented Kalman Filter*)
 ultra-violet radiation 287, 289
 uncertainty 160, 161, 163
 under-actuated 56, 99, 120, 121, 195, 229
 –, robot, manipulator 56, 210, 240, 241
 –, system 120
 unicycle, model 111
 unified imaging model 344, 565

- Unimation Inc. 2
 - unit
 - , inertial measurement (IMU) 40, 87, 577
 - , photometric 291
 - , quaternion 44, 45, 47, 55, 58, 499
 - , derivative 64
 - , interpolation 76
 - , normalization 50
 - , radiometric 291
 - , twist 30, 48, 52, 54
 - unmanned aerial vehicle (UAV) 114
 - unscented Kalman Filter (UKF) 184
 - utricle 83
-
- ## V
- VaMoRs system (autonomous van) 122
 - vanishing point 321, 328
 - variable, Gaussian random 636
 - Vaucanson's duck 1
 - vector 17, 587
 - , addition 587
 - , approach 40
 - , bound 17
 - , coordinate 17–19, 587, 595, 604
 - , distortion 406
 - , error 628
 - , feature 432, 434
 - , field 619
 - , gravity 84, 263
 - , moment 30, 47, 52, 351, 596
 - , normal 40
 - , of locally aggregated descriptors (VLAD) 456
 - , orientation 40
 - , point xxix, 17, 22
 - , Rodrigues 42
 - , rotation 30
 - , scalar function of 618
 - , singular 592
 - , space 587
 - , twist 30, 31, 47
 - , vector function of 618
 - , velocity 64, 230
 - vectorizing 493
 - vehicle
 - , aerial 121
 - , autonomous 7, 96
 - , surface (ASV) 96
 - , underwater (AUV) 96
 - , Braitenberg 126
 - , car-like 99, 100
 - , configuration 100
 - , coordinate system 100
 - , differentially-steered 99, 109
 - , frame 100
 - , micro air (MAV) 114
 - , mobile robot 3, 95, 99, 573
 - , model 107
 - , omnidirectional 112
 - , orientation 101, 108, 575
 - , path 103, 105, 109
 - , underwater 121
 - , unmanned aerial (UAV) 96, 114
 - , velocity 101
 - , wheeled 97, 99
 - velocity 251, 275
 - , angular 50, 52, 64, 68, 70, 79, 80, 155, 233, 636
 - , time-varying 68
 - , vector 66
 - , camera 542–544, 547, 551, 552, 556–559, 567
 - , control 102, 257, 261
 - , feedforward 262
 - , loop 257, 261
 - , coupling torque 264
 - , discontinuity 78
 - , ellipse, ellipsoid 235, 236, 244
 - , end-effector 229, 230
 - , joint 229, 230
 - , kinematics 229
 - , linear 52, 68
 - , maximum 72
 - , peak 72
 - , rotational 63, 65, 69
 - , spatial 64, 65, 69, 231, 232, 239, 542, 546, 573
 - , translational 63, 65, 69
 - , twist 65, 247
 - , vector 64, 230
 - , vehicle 101
 - vestibular system 80, 83, 546
 - via point 74
 - view
 - , field of 327, 336, 338, 339, 347, 348, 487, 546, 559, 572
 - , fronto-parallel 510, 511, 541
 - , road map 367
 - , satellite 367
 - vignetting 364
 - viscous friction coefficient 252
 - vision 6
 - , animal 285
 - , human 331
 - , robotic 6
 - , stereo 479, 488, 491, 503, 507, 509, 524
 - visual
 - , flux 287
 - , odometry (VO) 13, 520–522
 - , servo control 535
 - , servoing (see *servoing, visual*)
 - , simultaneous localization and mapping (VSLAM) 184, 498
 - , vocabulary 515
 - , word 515
 - VLAD (see *vector of locally aggregated descriptors*)
 - VO (see *visual odometry*)
 - von Mises distribution 156
 - Voronoi
 - , cell 137
 - , diagram 136, 137, 399
 - , roadmap 137
 - , tessellation 137
 - VSLAM (see *visual simultaneous localization and mapping*)
-
- ## W
- WAAS (see *wide area augmentation system*)
 - walking robot 221
 - warping 336, 345, 404–406, 502, 510, 513
 - waypoint 157
 - white
 - , balance, balancing 308
 - , D_{65} 304, 305, 306, 312
 - , definition 306

–, equal-energy 305
 –, point 302
 Wide Area Augmentation System (WAAS) 153

Wien's

–, approximation 314
 –, displacement law 288

window, convolution 376

world coordinate frame 18, 79

wrench 65, 69, 244, 245, 263, 269

–, ellipsoid 245
 –, end-effector 244, 245
 –, transformation 244

wrist 208

–, coordinate frame 203
 –, robot 196, 215
 –, singularity 208, 215
 –, spherical 199, 205, 207

X

Xbox 508

XY/Z-partitioned IBVS (image-based visual servo) 565

XYZ

–, CIE (Commission Internationale de l'Éclairage) primary 300

–, color

–, matching function 300

–, space 301, 312

–, roll-pitch-yaw angle 38, 214, 232

–, tristimulus value 304

Y

yaw angle 37

yaw rate 101, 163

$Y_C C_R$ color space 303, 311

Yoshikawa's manipulability measure 236

YUV color space 303, 311

YXZ roll-pitch-yaw angle 481

Z

zero-angle configuration 197

zero crossing detector 387

ZNCC similarity measure 389, 390, 461, 484, 485, 489, 514

zoom lens 327

ZSSD similarity measure 389, 530

ZYX roll-pitch-yaw angle 37

ZYZ Euler angles 36