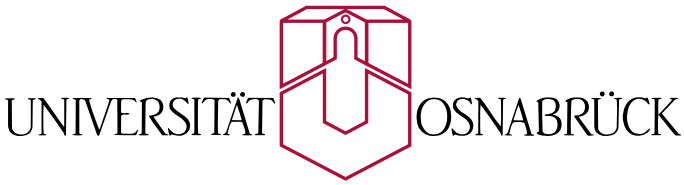
6D SLAM – Simultaneous 6 D.O.F. Localization and 3D Mapping



Andreas Nüchter, Kai Lingemann, Joachim Hertzberg
Department of Mathematics/Computer Science
Institute of Computer Science
Knowledge-Based Systems Research Group
University of Osnabrück
http://www.informatik.uni-osnabrueck.de/kbs/

June 15, 2009

Documentation

This document describes the algorithms for 6D SLAM – Simultaneous 6 D.O.F. Localization and 3D Mapping system. 6D SLAM with mobile robots considers six dimensions for the robot pose, namely the x, y and z coordinates and the roll, yaw and pitch angles. Robot motion and localization on natural surfaces, e.g., driving with a mobile robot outdoor, must necessarily regard these degrees of freedom.

1 Range Image Registration and Robot Relocalization

Multiple 3D scans are necessary to digitalize environments without occlusions. To create a correct and consistent model, the scans have to be merged into one coordinate system. This process is called registration. If robot carrying the 3D scanner were precisely localized, the registration could be done directly based on the robot pose. However, due to the unprecise robot sensors, self localization is erroneous, so the geometric structure of overlapping 3D scans has to be considered for registration.

The following method for registration of point sets is part of many publications, so only a short summary is given here. The complete algorithm was invented in 1992 and can be found, e.g., in [2]. The method is called *Iterative Closest Points (ICP) algorithm*.

Given two independently acquired sets of 3D points, M (model set, $|M| = N_m$) and D (data set, $|D| = N_d$) which correspond to a single shape, we aim to find the transformation consisting of a rotation $\boldsymbol{R}$ and a translation $\boldsymbol{t}$ which minimizes the following cost function:

$$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} \left|\left| \boldsymbol{m}_i - (\boldsymbol{R}\boldsymbol{d}_j + \boldsymbol{t}) \right|\right|^2 . \tag{1}$$

$w_{i,j}$ is assigned 1 if the i-th point of M describes the same point in space as the j-th point of D. Otherwise $w_{i,j}$ is 0. Two things have to be calculated: First, the corresponding points, and second, the transformation ($\boldsymbol{R}$, $\boldsymbol{t}$) that minimize $E(\boldsymbol{R}, \boldsymbol{t})$ on the base of the corresponding points.

The ICP algorithm calculates iteratively the point correspondences. In each iteration step, the algorithm selects the closest points as correspondences and calculates the transformation $(\boldsymbol{R}, \boldsymbol{t})$ for minimizing equation (1). The assumption is that in the last iteration step the point correspondences are correct. Besl et al. prove that the method terminates in a minimum [2]. However, this theorem does not hold in our case, since we use a maximum tolerable distance $d_{\max}$ for associating the scan data. Such a threshold is required, given that the 3D scans overlap only partially. Fig. 1 (top) shows three frames, i.e., iteration steps, of the ICP algorithm. The bottom part shows the start poses (x, z, θ_y) from which a correct matching is possible, here with only three degrees of freedom.

1.1 Calculation of the rotation and translation

In every iteration the optimal tranformation ($\boldsymbol{R}$, $\boldsymbol{t}$) has to be computed. Eq. (1) can be reduced to

$$E(\boldsymbol{R}, \boldsymbol{t}) \quad \propto \quad \frac{1}{N} \sum_{i=1}^{N} \left|\left| \boldsymbol{m}_i - (\boldsymbol{R}\boldsymbol{d}_i + \boldsymbol{t}) \right|\right|^2 , \tag{2}$$

with $N = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j}$, since the correspondence matix can be represented by a vector containing the point pairs.

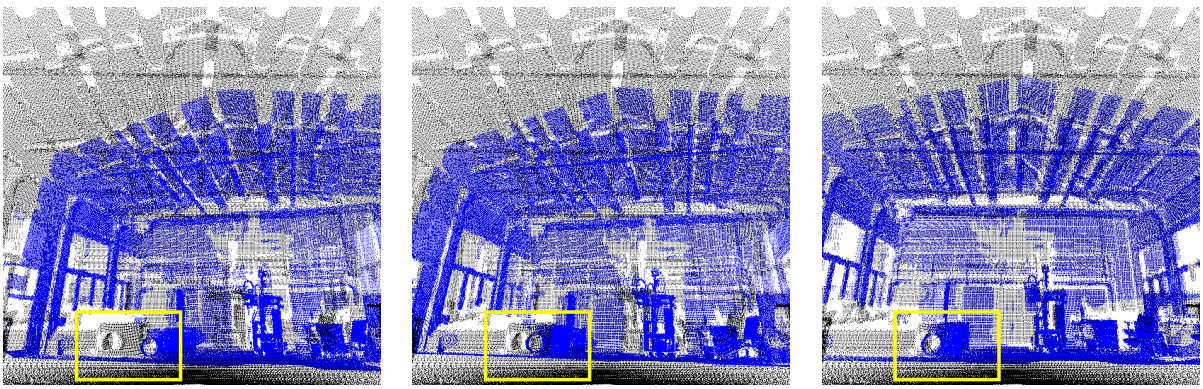


Figure 1: Left: Initial odometry based pose of two 3D scans. Middle: Pose after five ICP iterations. Right: final alignment, pairwise matching.

Four methods are known to minimize eq. (2) [3]. The 6D SLAM system uses the following one, based on singular value decomposition (SVD), is robust and easy to implement, thus we give a brief overview of the SVD-based algorithms. It was first published by Arun, Huang and Blostein [1]. The difficulty of this minimization problem is to enforce the orthonormality of matrix $\boldsymbol{R}$. The first step of the computation is to decouple the calculation of the rotation $\boldsymbol{R}$ from the translation $\boldsymbol{t}$ using the centroids of the points belonging to the matching, i.e.,

$$\boldsymbol{c}_m = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{m}_i, \qquad \boldsymbol{c}_d = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{d}_j \tag{3}$$

and

$$M' = \{\boldsymbol{m}'_i = \boldsymbol{m}_i - \boldsymbol{c}_m\}_{1,\ldots,N}, \tag{4}$$

$$D' = \{\boldsymbol{d}'_i = \boldsymbol{d}_i - \boldsymbol{c}_d\}_{1,\ldots,N}. \tag{5}$$

After replacing (3), (4) and (5) in the error function, $E(\boldsymbol{R},\boldsymbol{t})$ eq. (2) becomes:

$$E(\boldsymbol{R},\boldsymbol{t}) \propto \frac{1}{N}\sum_{i=1}^{N}||\boldsymbol{m}'_i - \boldsymbol{R}\boldsymbol{d}'_i - \underbrace{(\boldsymbol{t} - \boldsymbol{c}_m + \boldsymbol{R}\boldsymbol{c}_d)}_{=\tilde{\boldsymbol{t}}}||^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}||\boldsymbol{m}'_i - \boldsymbol{R}\boldsymbol{d}'_i||^2 \tag{6a}$$

$$-\frac{2}{N}\tilde{\boldsymbol{t}}\cdot\sum_{i=1}^{N}(\boldsymbol{m}'_i - \boldsymbol{R}\boldsymbol{d}'_i) \tag{6b}$$

$$+\frac{1}{N}\sum_{i=1}^{N}\left|\left|\tilde{\boldsymbol{t}}\right|\right|^2. \tag{6c}$$

In order to minimize the sum above, all terms have to be minimized. The second sum (6b) is zero, since all values refer to centroid. The third part (6c) has its minimum for $\tilde{\boldsymbol{t}} = \mathbf{0}$ or

$$\boldsymbol{t} = \boldsymbol{c}_m - \boldsymbol{R}\boldsymbol{c}_d. \tag{7}$$

Therefore the algorithm has to minimize only the first term, and the error function is expressed in terms of the rotation only:

$$E(\boldsymbol{R}, \boldsymbol{t}) \propto \sum_{i=1}^{N} ||\boldsymbol{m}'_i - \boldsymbol{R}\boldsymbol{d}'_i||^2 . \tag{8}$$

Theorem: The optimal rotation is calculated by $\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T$. Herby the matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ are derived by the singular value decomposition $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T$ of a correlation matrix $\boldsymbol{H}$. This 3×3 matrix $\boldsymbol{H}$ is given by

$$\boldsymbol{H} = \sum_{i=1}^{N} \boldsymbol{m}_i'^T \boldsymbol{d}'_i = \begin{pmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix}, \tag{9}$$

with $S_{xx} = \sum_{i=1}^{N} m'_{ix} d'_{ix}$, $S_{xy} = \sum_{i=1}^{N} m'_{ix} d'_{iy}$, ... The analogous algorithm is derived directly from this theorem.

Proof: Since rotation is length preserving, i.e., $||\boldsymbol{R}\boldsymbol{d}'_i||^2 = ||\boldsymbol{d}'_i||^2$ the error function (8) is expanded

$$E(\boldsymbol{R}, \boldsymbol{t}) \propto \sum_{i=1}^{N} ||\boldsymbol{m}'_i||^2 - 2\sum_{i=1}^{N} \boldsymbol{m}'_i \cdot \boldsymbol{R}\boldsymbol{d}'_i + \sum_{i=1}^{N} ||\boldsymbol{d}'_i||^2 .$$

The rotation affects only the middle term, thus it is sufficient to maximize

$$\sum_{i=1}^{N} \boldsymbol{m}'_i \cdot \boldsymbol{R}\boldsymbol{d}'_i \;=\; \sum_{i=1}^{N} \boldsymbol{m}_i'^T \boldsymbol{R}\boldsymbol{d}'_i. \tag{10}$$

Using the trace of a matrix, (10) can be rewritten to obtain

$$\mathrm{tr}\left(\sum_{i=1}^{N} \boldsymbol{R}\boldsymbol{d}'_i \boldsymbol{m}_i'^T\right) = \mathrm{tr}\left(\boldsymbol{R}\boldsymbol{H}\right),$$

With $\boldsymbol{H}$ defined as in (9). Now we have to find the matrix $\boldsymbol{R}$ that maximizes $\mathrm{tr}\left(\boldsymbol{R}\boldsymbol{H}\right)$. Assume that the singular value decomposition of $\boldsymbol{H}$ is

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T,$$

with $\boldsymbol{U}$ and $\boldsymbol{V}$ orthonormal 3×3 matrices and $\boldsymbol{\Lambda}$ a 3×3 diagonal matrix without negative elements. Suppose

$$\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T.$$

$\boldsymbol{R}$ is orthonormal and

$$\begin{aligned}\boldsymbol{R}\boldsymbol{H} &= \boldsymbol{V}\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T \\ &= \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T\end{aligned}$$

is a symmetric, positive definite matrix. Arun, Huang and Blostein provide a lemma to show that

$$\mathrm{tr}\,(\boldsymbol{R}\boldsymbol{H}) \geq \mathrm{tr}\,(\boldsymbol{B}\boldsymbol{R}\boldsymbol{H})$$

for any orthonormal matrix $\boldsymbol{B}$. Therefore the matrix $\boldsymbol{R}$ is optimal. Prooving the lemma is straightforward using the Cauchy-Schwarz [1]. Finally, the optimal translation is calculated as (cf. eq. (6c) and (7))

$$\boldsymbol{t} = \boldsymbol{c}_m - \boldsymbol{R}\boldsymbol{c}_d.$$

2 ICP-based 6D SLAM

To match two 3D scans with the ICP algorithm it is necessary to have a sufficient starting guess for the second scan pose.

- Extrapolate the odometry readings to all six degrees of freedom using previous registration matrices. The change of the robot pose $\Delta\boldsymbol{P}$ given the odometry information $(x_n, z_n, \theta_{y,n})$, $(x_{n+1}, z_{n+1}, \theta_{y,n+1})$ and the registration matrix $\boldsymbol{R}(\theta_{x,n}, \theta_{y,n}, \theta_{z,n})$ is calculated by solving:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \\ z_{n+1} \\ \theta_{x,n+1} \\ \theta_{y,n+1} \\ \theta_{z,n+1} \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \\ z_n \\ \theta_{x,n} \\ \theta_{y,n} \\ \theta_{z,n} \end{pmatrix} + \left(\begin{array}{c|ccc} \boldsymbol{R}(\theta_{x,n}, \theta_{y,n}, \theta_{z,n}) & \multicolumn{3}{c}{\boldsymbol{0}} \\ \hline \boldsymbol{0} & 1 & 0 & 0 \\ & 0 & 1 & 0 \\ & 0 & 0 & 1 \end{array}\right) \cdot \underbrace{\begin{pmatrix} \Delta x_{n+1} \\ \Delta y_{n+1} \\ \Delta z_{n+1} \\ \Delta\theta_{x,n+1} \\ \Delta\theta_{y,n+1} \\ \Delta\theta_{z,n+1} \end{pmatrix}}_{\Delta\boldsymbol{P}}. \quad (11)$$

  Therefore, calculating $\Delta\boldsymbol{P}$ requires a matrix inversion. Finally, the 6D pose $\boldsymbol{P}_{n+1}$ is calculated by

$$\boldsymbol{P}_{n+1} = \Delta\boldsymbol{P} \cdot \boldsymbol{P}_n$$

  using the poses' matrix representations.

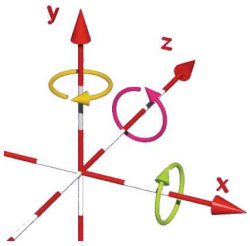


Figure 2: Left handed coordinate system.

3 Variable Correspondences

$(\boldsymbol{R}, \boldsymbol{t})$	`double alignxf[16]`	Transformation Matrix
$\boldsymbol{m}_i$, $\boldsymbol{d}_i$	`class PtPair`	Point Pair
$\boldsymbol{c}_m$, $\boldsymbol{c}_d$	`double cm[3], cd[3]`	Centroids
$\boldsymbol{m}'_i$, $\boldsymbol{d}'_i$	`double** m, d`	Centered Point Pairs
$\boldsymbol{H}$, $\boldsymbol{U}$, $\boldsymbol{\Lambda}$, $\boldsymbol{V}$	`Matrix`	SVD Matrices
$\boldsymbol{R}$	`double transMat[16]`	Pose as Matrix
(x_n, y_n, z_n)	`double rPos[3]`	Position of n-th 3D Scan
$(\theta_{x,n}, \theta_{y,n}, \theta_{z,n})$	`double rPostheta[3]`	Rotation of n-th 3D Scan

4 File Formats and Units

The coordinate system is left handed, with the y axis pointing upwards, and the depth axis z (cf. Figure 2). Input and output files are:

1. The 3D scan files (`scanXXX.3d`)[1] have to be of the following structure:
   The first line contains the scan's resolution (w x b), followed by lines of data points (x, y, z).

2. The pose files (`scanXXX.pose`)[1] associated with each 3D scan contain information of the estimated pose of the respective scan as given by, e.g., odometry. The first line contains the 3 translatorial positions (x, y, z), the second the rotations pitch, yaw and roll (θ_x, θ_y, θ_z around the respective axis) in deg.
   Values that are not estimated by the robot (odometry) can be set to 0 and are extrapolated as described by Eq. (11).

3. The SLAM program generated files `scanXXX.frames`, consisting of the transformations computed from the scan matching. Each line contains a 4×4 OpenGL-style matrix. The very last matrix is the final transformation for registering the scan into the common coordinate system.

[1] stating with `XXX` = 000, 001,..., until no more files are found in the specified directory.

The matrix is stored in the following format:

$(R[1,1], R[1,2], 0, R[1,3], 0, R[2,1], R[2,2], R[2,3], 0, R[3,1], R[3,2], R[3,3], 0, t[1], t[2], t[3], 1)$,

with $R[x, y]$ the (x, y)-th entry of the rotation matrix $\boldsymbol{R}$, and $t[x]$ the x-th translation component of $\boldsymbol{t}$.

5 Requirements

All executables can be compiled and used both with Linux and Windows.

Linux: The system was developed and tested under Linux 9.1 with the g++ compiler version 3.3.3. As additional library, `OpenGL` and `glut` have to be installed, which should be included in your Linux distribution (tested with freeglut 2.2.0-78).
To compile, type in `make` in the main directory. The executables are generated in the `./bin` directory.

Windows: The system was tested with the C++ compiler from Microsoft Visual Studio.NET 2005. To compile, load the respective project file (`.sln`) from the directory `.\Visual_Studio_Projects\`. The executables are generated in the respective `Debug` or `Release` directories, depending on your compiler settings.
Precompiled versions can be found in the `./bin` directory, too. If moving the executables, take care about the `glut` directory as well.

6 Usage

For a detailed explanation about the programs' usage, just start the respective binary. Both applcations can be configured by a set of command line parameters, which are explained when starting the program as mentioned above.

Especially, take care of the reduction parameters `-r`/`-R` of the SLAM system: Without using one of those, the registration is being slowed down tremendously due to taking *all* data points as input. Other potentially critical parameters are the maximal distance of points that may form corresponding point pairs (matrix entries w_{ij}, parameter `-d`), as well as the maximal range distance of points used for scan matching or displaying (`-m`), especially used for eliminating outliers (i.e., data points with the maximal range distance of the range finder).

References

[1] K. S. Arun, T. S. Huang, and S. D. Blostein. Least square fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698 – 700, 1987.

[2] P. Besl and N. McKay. A method for Registration of 3–D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239 – 256, February 1992.

[3] A. Lorusso, D. Eggert, and R. Fisher. A Comparison of Four Algorithms for Estimating 3-D Rigid Transformations. In *Proceedings of the 4th British Machine Vision Conference (BMVC '95)*, pages 237 – 246, Birmingham, England, September 1995.