Interactive 3D Scan-Matching Using RGB-D Data*

Pedro Vieira Rodrigo Ventura
Institute for Systems and Robotics
Instituto Superior Técnico
TULisbon, Portugal
pedro.s.vieira@ist.utl.pt, rodrigo.ventura@isr.ist.utl.pt

Abstract

Several methods have been proposed in the literature to address the problem of automatic 3D reconstruction from depth data. Most methods rely on the minimization of the matching error among individual depth frames. However, ambiguity in sensor data often leads to erroneous matching (due to local minima), hard to cope with in a purely automatic approach. This paper proposes a semi-automatic approach, denoted interactive mapping, involving a human operator in the process of detecting and correcting erroneous matches. Instead of allowing the operator complete freedom in correcting the matching in a frame by frame basis, the proposed method constrains human intervention along the degrees of freedom with most uncertainty. This paper is targeted to 3D reconstruction from RGB-D data, such as the one provided by the Kinect sensor. The user is able to translate and rotate individual RGB-D point clouds, with the help of a force field-like reaction to the movement of each point cloud. Some preliminary results are presented, illustrating the advantages of the method.

1. Introduction

The problem of scan-matching has been an active field of research for a long time. Methods to address the problem have been proposed and used in many applications, in particular for 3D mapping of an environment (see [7] for a review). These methods vary both in terms of sensors used (e.g., sonars [2], LIDAR [5], vision [6], and more recently the Kinect [4]), and in methodologies (e.g., probabilistic [3], scan matching [5]). However, most of these methods are prone to local minima, originated, for instance, from ambiguity or from locally periodic patterns in the environment.

In this paper we propose an alternative approach where we consider the human-in-the-loop of the scan matching process. In particular, the user is invited to interact with the matching process, by adjusting the match of individual

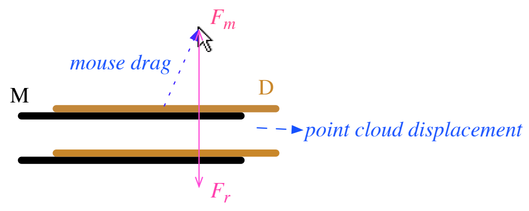

Figure 1. Top view of 2 corridors. The scan D is adjusted such that the reaction force F_r, computed from the cost function gradient, balances the force F_m imposed by the mouse drag.

3D pairs of scans. This adjustment is however constrained by favoring adjustments along the directions of greater ambiguity. Take for instance a case of pairs of identical scans taken from an homogeneous corridor (Fig. 1): the system favors the adjustment of the scans along the corridor, while disfavoring movements orthogonal to the corridor. The proposed method is based on a graphical user interface (GUI), where the user interacts with the system using a common computer interface (mouse and keyboard). Consider a pair of range scans denoted M (for model) and D (for data). Fig. 1 illustrates the situation for the corridor example. When the user drags one of the range scans, say scan D, using the mouse, a corresponding virtual force F_m is produced (proportional to the drag vector). Opposing it, a reaction force F_r is computed based in the match between scans M and D. Considering the scan matching cost function as a potential function, the symmetric of its gradient with respect to the shift of scan D can be seen as a reaction force F_r. This reaction force "attracts" scan D towards M, along the direction of less ambiguity. Scan D is then moved such that both forces become balanced, $F_m + F_r = 0$. In the corridor example of Fig. 1, the reaction force is (mostly) orthogonal to the corridor axis.

This paper is organized as follows. In Section 2 the method for 3D interactive mapping is proposed, followed by Section 3 showing the preliminary results based on the

*This work was supported by the FCT projects [PEst-OE/EEI/LA0009/2011] and [PTDC/EIA-CCO/113257/2009].

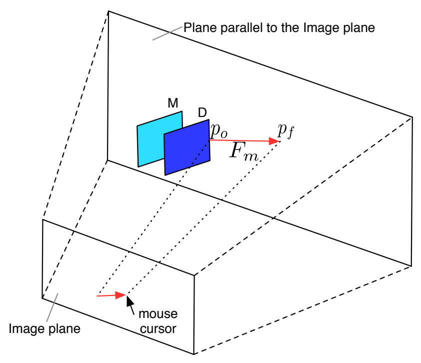

Figure 2. 3D representation of the mouse force, defined by p_o and p_f. Force is applied to point cloud D.

user experience with the alignment. Finally, Section 4 draws some conclusions and discuss future work directions.

2. 3D Interactive Alignment

Consider two point clouds, $M = \{m_k\}$ and $D = \{d_k\}$ and an initial rigid transformation (R,t) which align D with M. This transformation can be a default value (e.g. Identity matrix) or computed with an alignment algorithm, like ICP algorithm (see [1]). By applying this transformation to D, a transformed point cloud, D' is obtained. The two point clouds (M and D') are then presented in a viewer (GUI) for alignment analysis. Then, the user interacts with the viewer using common computer mouse and keyboard, and apply either translations or rotations to D'. These actions are carried out separately using a designated key to choose which mode to use.

Point clouds are aligned by balancing a virtual force caused by a mouse drag (mouse force) with a reaction force produced by a potential field. The drag is defined by two points: p_o, the mouse pointer position where the user clicked on the mouse, and p_f, the current pointer position. In 3D, we only have access to the coordinates of the pixel of the mouse current position in the image plane, therefore the mouse force is defined in a 3D plane parallel to the image plane (Fig. 2). The cost function that minimizes the distance between p_o and p_f, being responsible for creating the mouse force, is given by:

$$J_m = \frac{1}{2}\|p_f - [R(p_o - c) + c + t]\|^2 \qquad (1)$$

where R is a 3x3 matrix representing the 3D rotation of the point cloud, t a translation and c is the center of rotation. Rotations are made with respect to this center of rotation.

The cost function that minimizes the distances between corresponding points of the two point clouds, being responsible for originating the Potential field, and therefore for creating the reaction force is given by:

$$J_r = \frac{1}{2}\sum_{k=1}^{N}\|m_k - [R(d_k - c) + c + t]\|^2 \qquad (2)$$

where $\{m_k\}$ and $\{d_k\}$ are pairs of closest points from M and D', and N is the number of these pairs. This cost function is identical to the one used in ICP. The closest points are computed in the same fashion as in ICP, and only the pairs sufficiently close are considered.

2.1. Translations

In this mode, the alignment consists in a translation by t. Thus R is the identity, and function cost (1) is simplified. The mouse force F_m is computed from the gradient of the cost function (1) with respect to the translation t:

$$F_m = -k_m \nabla_t \left. J_m\right|_{R=I} = k_m\ (p_f - p_o - t) \qquad (3)$$

where k_m is the proportionality constant.

Opposing this force, a reaction force F_r is computed from the gradient of the cost function (2) with respect to the translation t:

$$F_r = -k_r \nabla_t \left. J_r\right|_{R=I} = k_r \sum_{k=1}^{N}(m_k - d_k - t) \qquad (4)$$

where k_r is the proportionality constant,

To find the adjustment that balances the mouse and the reaction forces, translations are iteratively performed, since each time the point cloud D' moves, the correspondences among points may change. So, for each iteration, a translation t is computed by solving the equation

$$F_m + F_r = 0 \qquad (5)$$

with respect to t. This equation has the following algebraic closed form solution:

$$t = \frac{k_m(p_f - p_o) + k_r \sum_{k=1}^{N}(m_k - d_k)}{k_m + Nk_r} \qquad (6)$$

The scan adjustment results from the following algorithm:

1. Compute translation t according to (6);
2. Compute the new correspondences $\{m_k\}$ and $\{d_k\}$ from scans M and D';
3. Unless the correspondences are the same, go to step 1).

The obtained $t = [t_x\ t_y\ t_z]^T$ corresponds to the homogeneous transformation:

$$T_t = \left[\begin{array}{c|c} I_{3\times3} & t \\ \hline 0 & 1 \end{array}\right] \qquad (7)$$

where $I_{3\times3}$ is a 3×3 identity matrix.

2.2. Rotations

Rotation mode inflict a rotation of point cloud D', with respect to the center of mass c. The center of mass of a scan is set to its centroid. When the user clicks on the point cloud D' and attempts to drag it, a force F_m is created using (1), for $t = 0$. However, unlike translations, the balance is formulated here in terms of virtual torques.

The mouse torque τ_m is the gradient of the cost function (1) computed around rotation R. We consider infinitesimal rotations α, β and γ, along axis x, y, and z.

$$\tau_m = -k_m \nabla_{\alpha,\beta,\gamma} J_m|_{t=0,\alpha=\beta=\gamma=0} \tag{8}$$

The opposing torque τ_r is the gradient of the cost function (2) with respect to α, β and γ:

$$\tau_r = -k_r \nabla_{\alpha,\beta,\gamma} J_r|_{t=0,\alpha=\beta=\gamma=0} \tag{9}$$

Note that Euler angles suffer from singularities, however, the rotation is parametrized using a rotation matrix, while the Euler angles are only used for computing derivatives. Thus, derivatives in (8) and (9) are done around the point $\alpha = 0$, $\beta = 0$ and $\gamma = 0$. As in the case of translations, the point cloud D' is iteratively rotated until convergence of the correspondences is reached. In each iteration, due the non-linearity of the system, a suitable solver is used to balance the two torques:

$$\tau_m + \tau_r = 0 \tag{10}$$

Balancing the torques consists in solving the following system of equations with respect to R:

$$\mathrm{tr}\left[\left(k_m r_i \left(p'_f\right)^T + k_r \sum_{k=1}^{N} d'_k \left(m'_k\right)^T\right) A_l R\right] = 0 \tag{11}$$

$$R^T R = I\,, \quad l = 1,2,3 \tag{12}$$

where $r_i = (p_o - c)$, $p'_f = (p_f - c)$, $d'_k = (d_k - c)$, $m'_k = (m_k - c)$ and

$$A_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad A_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad A_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The scan adjustment follows a similar algorithm as in the case of translations:

1. Compute rotation R using a suitable solver to resolve (11) and (12)

Step 2 and 3 are exactly the same as in translations.

A rotation transformation is then created using the matrix R computed,

$$T_{\alpha,\beta,\gamma} = \left[\begin{array}{c|c} R & b \\ \hline 0 & 1 \end{array}\right] \tag{13}$$

$$b = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [I - R]\, c \tag{14}$$

Note that translation b accounts for the fact that the rotation is performed with respect to center c, rather than to the origin.

2.3. Restricted rotation mode

An alternative mode for rotating point clouds is to make the rotations around an axis perpendicular to the plane parallel to the image plane, and centered in the center of mass projected on this plane. The Degrees of Freedom (DoFs) are reduce from 3 to 1. The Rotation matrix is given by the Rodrigues' rotation formula, which says that given a unit vector $u = (u_x, u_y, u_z)$, where $u_x^2 + u_y^2 + u_z^2 = 1$, the matrix for a rotation by an angle θ about an axis in the direction of u is:

$$R = I\cos\theta + \sin\theta\, U_1 + (1 - \cos\theta)\, U_2 \tag{15}$$

where

$$U_1 = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \quad U_2 = \begin{bmatrix} u_x^2 & u_x u_y & u_x u_z \\ u_x u_y & u_y^2 & u_y u_z \\ u_x u_z & u_y u_z & u_z^2 \end{bmatrix}$$

The mouse torque τ_m is the gradient of the cost function (1) with respect to θ:

$$\tau_m = -k_m \nabla_\theta J_m|_{t=0} \tag{16}$$

The opposing torque τ_r is the gradient of the cost function (2) with respect to θ:

$$\tau_r = -k_r \nabla_\theta J_r|_{t=0} \tag{17}$$

The balance of torques (10) has a closed form solution that can be obtained using (16) and (17):

$$\tan(\theta) = \frac{k_m r_i^T U_1^T p'_f + k_r \sum_{k=1}^{N} (d'_k)^T U_1^T m'_k}{k_m r_i^T U'_2 p'_f + k_r \sum_{k=1}^{N} (d'_k)^T U'_2 m'_k} \tag{18}$$

where $U'_2 = (I - U_2)^T$. However, this only allows us to compute θ up to a π congruence. To resolve this ambiguity, which is caused by the existence of two solutions π radians apart, one has to determine which one corresponds to a stable solution. This can be easily determined from the sign of the derivative of the total torque $\tau = \tau_m + \tau_r$,

$$\frac{\partial \tau}{\partial \theta} = -H_1 \cos\theta - H_2 \sin\theta \tag{19}$$

where

$$H_1 = k_m r_i^T U'_2 p'_f + k_r \sum_{k=1}^{N} (d'_k)^T U'_2 m'_k \tag{20}$$

$$H_2 = k_m r_i^T U_1^T p'_f + k_r \sum_{k=1}^{N} (d'_k)^T U_1^T m'_k \tag{21}$$

A solution is stable if and only if the sign of this derivative is negative. A positive derivative implies that a small perturbation in θ will swing the point cloud π radians towards the other solution. Note that (19) has opposite signs for angles θ and $\theta + \pi$, and therefore there is always a single negative solution.

The scan adjustment follows a similar algorithm as in the case of translations:

1. Compute rotation θ according to (18), choosing the solution θ or $\theta + \pi$ with negative derivative (19);

Step 2 and 3 are exactly the same as in translations.

A rotation transformation is then created using the value θ, and equations (13), (14) and (15).

3. Preliminary Results

Figure 3 illustrates the result of applying interactive alignment to a badly aligned collection of point clouds: (1) shows this collection after an initial pass through ICP. The resulting alignment is visibly erroneous, due to local minima. A user was then asked to employ the proposed method to interactively align the point clouds: (3) and (4) show an arbitrary pair of (consecutive) point clouds, before and after the alignment. This alignment demanded for a translation and a rotation: (2) shows the final result after the user aligned, in a pair by pair basis, all pairs of frames.

In qualitative terms, users have found the proposed interactive method useful. They showed preference in making rotations restricted to a plane, because these are easier to control and more intuitive. However, when using forces, the other rotation mode performs better when the two point clouds are nearly aligned, due the fact of having three DoFs.

4. Conclusions and Future Work

In this paper we have proposed a method using virtual forces with the purpose of helping users to adjust the alignment of 3D point clouds. The 3D interactive alignment with the use of forces proved to be a valuable aid in the correction of the alignment. As future work we propose making a mode detailed user study of the proposed method, and apply it to larger 3D environments.

References

[1] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992.

[2] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE J. Robot. Automat.*, 3(3):249–265, June 1987.

[3] A. Elfes. *Occupancy grids: A probabilistic framework for robot perception and navigation*. PhD thesis, Carnegie Mellon University, 1989.

[4] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3d modeling of indoor environments. In *The 12th International Symposium on Experimental Robotics (ISER)*, 2010.

[5] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous robots*, 4(4):333–349, 1997.

[6] D. Murray and C. Jennings. Stereo vision based mapping and navigation for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA'97)*, volume 2, pages 1694–1699, April 1997.

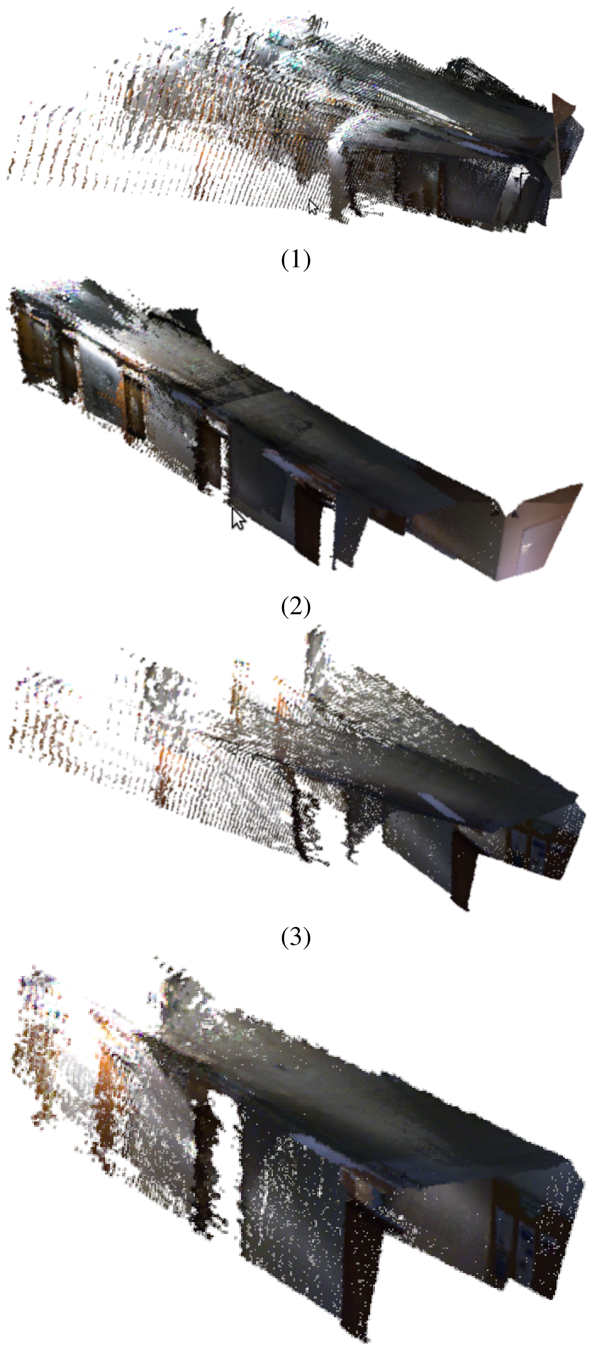

Figure 3. From top to bottom: (1) initial 3D map after applying ICP to the raw RGB-D data; (2) final map after interactive alignment; (3) a pair of point clouds from the initial map; and (4) the same pair after interactive alignment.

[7] S. Thrun. *Exploring artificial intelligence in the new millennium*, chapter Robotic mapping: A survey, pages 1–35. Morgan Kaufmann, 2002.