

## A Theory of Minimal 3D Point to 3D Plane Registration and Its Generalization

Ramalingam, S.; Taguchi, Y.

TR2012-075 September 2012

### Abstract

Registration of 3D data is a key problem in many applications in computer vision, computer graphics and robotics. This paper provides a family of minimal solutions for the 3D-to-3D registration problem in which the 3D data are represented as points and planes. Such scenarios occur frequently when a 3D sensor provides 3D points and our goal is to register them to a 3D object represented by a set of planes. In order to compute the 6 degrees-of-freedom transformation between the sensor and the object, we need at least six points on three or more planes. We systematically investigate and develop pose estimation algorithms for several configurations, including all minimal configurations, that arise from the distribution of points on planes. We also identify the degenerate configurations in such registrations. The underlying algebraic equations used in many registration problems are the same and we show that many 2D-to-3D and 3D-to-3D pose estimation/registration algorithms involving points, lines, and planes can be mapped to the proposed framework. We validate our theory in simulations as well as in three real-world applications: registration of a robotic arm with an object using a contact sensor, registration of planar city models with 3D point clouds obtained using multi-view reconstruction, and registration between depth maps generated by a Kinect sensor

*International Journal of Computer Vision*

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.



# A Theory of Minimal 3D Point to 3D Plane Registration and Its Generalization

Srikumar Ramalingam · Yuichi Taguchi

Received: 6 November 2011 / Accepted: 12 September 2012  
© Springer Science+Business Media New York 2012

**Abstract** Registration of 3D data is a key problem in many applications in computer vision, computer graphics and robotics. This paper provides a family of minimal solutions for the 3D-to-3D registration problem in which the 3D data are represented as points and planes. Such scenarios occur frequently when a 3D sensor provides 3D points and our goal is to register them to a 3D object represented by a set of planes. In order to compute the 6 degrees-of-freedom transformation between the sensor and the object, we need at least six points on three or more planes. We systematically investigate and develop pose estimation algorithms for several configurations, including all minimal configurations, that arise from the distribution of points on planes. We also identify the degenerate configurations in such registrations. The underlying algebraic equations used in many registration problems are the same and we show that many 2D-to-3D and 3D-to-3D pose estimation/registration algorithms involving points, lines, and planes can be mapped to the proposed framework. We validate our theory in simulations as well as in three real-world applications: registration of a robotic arm with an object using a contact sensor, registration of planar city models with 3D point clouds obtained using multi-view reconstruction, and registration between depth maps generated by a Kinect sensor.

**Keywords** 3D-to-3D registration · Pose estimation · Point-to-plane registration · Minimal solution · Correspondence problem

---

S. Ramalingam (✉) · Y. Taguchi  
Mitsubishi Electric Research Laboratories (MERL), Cambridge,  
MA, USA  
e-mail: [ramalingam@merl.com](mailto:ramalingam@merl.com)

## 1 Introduction and Previous Work

Pose estimation refers to the estimation of 6-degree-of-freedom (6-DoF) object pose using sensor measurements (e.g., images, 3D point clouds) and prior knowledge (e.g., a 3D model) of the object. This is achieved by registering the 3D data from the sensor to the known 3D model of the object. Such registration algorithms play a major role in numerous applications including object recognition, tracking, localization and mapping, augmented reality, and medical image alignment. Recent progress in the availability of 3D sensors such as Kinect (Shotton et al. 2011) at reasonable cost has further accelerated the need for such problems. The registration problem can generally be seen as two subproblems: a correspondence problem and a problem of pose estimation given the correspondence. Both of these problems are intertwined, and the solution of one depends on the other. This paper addresses the solution to both problems, although the major emphasis is on the second one.

Several 3D-to-3D registration scenarios are possible depending on the representation of the two 3D datasets: 3D points to 3D points, 3D lines to 3D planes, 3D points to 3D planes, etc. (Olsson et al. 2006). Iterative closest point (ICP) and its variants have been the gold standard in the last two decades (Besl and McKay 1992; Fitzgibbon 2003). These algorithms perform very well with a good initialization. Hence the main unsolved problem is the initial coarse registration. The registration of 3D lines to 3D planes and the registration of 3D points *with normals* to 3D planes were considered in Chen (1991), Grimson and Lozano-Pérez (1983). In contrast to their work, we register 3D points without normals to 3D planes. Note that the presence of normals makes the problem much simpler compared to the case without the normals. Recently, there have been several registration algorithms that focus on solving both the correspondence

and pose estimation (Enqvist et al. 2009; Tu et al. 1999; Li and Hartley 2007), primarily by casting the correspondence problem as a graph theoretical one. The correspondence problem maps to a class of NP-hard problems such as minimum vertex cover (Enqvist et al. 2009) and maximum clique (Tu et al. 1999). In this paper, we briefly explain the correspondence problem by formulating it as a maximum clique problem. Several variants of RANSAC (Fischler and Bolles 1981; Raguram et al. 2008) and voting based algorithms (Drost et al. 2010) have been developed for accurate and efficient solutions, which is a different line of research.

The main focus of this paper is on solving for the point-to-plane registration given the correspondence. Despite several existing results in 3D-to-3D registration problems, the registration of points to planes has received very little attention. However, in practice many registration problems can be efficiently solved by formulating them as point-to-plane. Iterative approaches exist for this problem (Chen and Medioni 1991; Olsson et al. 2006). In Olsson et al. (2006), the authors specifically mention that their algorithms had difficulties with point-to-plane registration and pointed out the need for a minimal solution. The minimal solution developed here provides a clear understanding of degenerate cases of the point-to-plane registration.

The development of minimal solutions in general has been beneficial in several vision problems (Kukelova and Pajdla 2007; Gao et al. 2003; Stewenius et al. 2005a, 2005b; Geyer and Stewenius 2007; Li and Hartley 2005). Minimal solutions have been proposed for several computer vision problems: auto-calibration of radial distortion (Kukelova and Pajdla 2007), perspective three point problem (Gao et al. 2003), the five point relative pose problem (Nistér 2003), the six point focal length problem (Stewenius et al. 2005a), the six point generalized camera problem (Stewenius et al. 2005b), the nine point problem for estimating para-catadioptric fundamental matrices (Geyer and Stewenius 2007), the nine point radial distortion problem (Li and Hartley 2005), the absolute pose with known vertical direction (Kukelova et al. 2010), and the relative pose with known gravity vector (Naroditsky et al. 2012). The last few years have seen the use of minimal problems in various applications (Snavely et al. 2006) and there are even unification efforts to keep track of all the existing solutions.<sup>1</sup> Minimal solutions have proven to be less noise-prone than non-minimal algorithms, and they have been quite useful in practice as hypothesis generators in hypothesize-and-test algorithms such as RANSAC (Fischler and Bolles 1981; Raguram et al. 2008).

## 1.1 Contributions

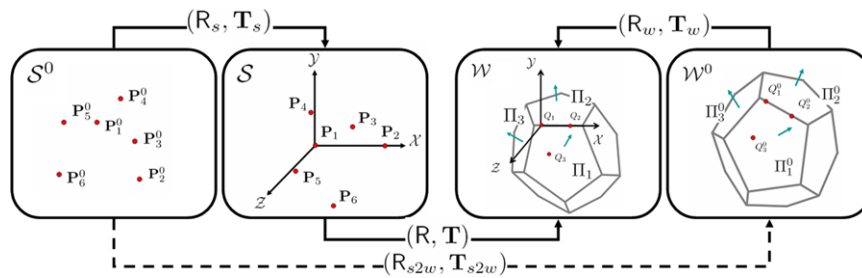
The main contribution of this paper is to systematically consider several cases in which we know the distribution of the points on the planes (how many points correspond to each plane), developing a customized pose estimation algorithm for each case. We denote each configuration as  $Points(a_1, a_2, \dots, a_n) \leftrightarrow Planes(n)$ , where  $n = \{3, 4, 5, 6\}$  is the number of distinct planes on which the points lie, and  $a_i$  is the number of points that lie on the  $i$ th plane. Although the general framework is the same, the underlying theory for each case is different and thereby leads to polynomial equations of different degrees. We will show later that this case-by-case analysis of point-to-plane registration not only helps to develop efficient algorithms, but also avoids degeneracy problems that arise in using an algorithm that is independent of the configurations. In addition to point-to-plane registration, we will show that several existing registration algorithms can be mapped to one of these different cases of point-to-plane registration. Many problems in multi-view geometry and registration appear different. However, the solutions to such problems can be highly interconnected. The differences between several problems in geometry can be seen from the underlying constraints and the number of variables. For example, one can classify the problems in geometry that compute rotation and translation into the following categories that are hierarchically more complex.

- 5 degrees of motion up to a scale (5-point algorithm: Nistér 2003)
- 6 degrees of motion (point-to-plane registration, 6-point non-central motion estimation: Stewenius et al. 2005b)
- 11 degrees of freedom up to a scale (3 view 4 point case: Nistér and Schaffalitzky 2006, 3 view 6 lines case)
- 12 degrees of freedom (minimal solution for generic calibration, trifocal algorithms for non-central cameras, etc.)

This paper falls in the second category where we compute 6 degrees of motion parameters. However, the underlying equations are simpler than the ones used in generalized motion estimation (Stewenius et al. 2005b).

To validate our theory we show an exhaustive set of simulations and three compelling real-world proof-of-concept experiments: registration of a robotic arm with an object using a contact sensor, registration of 3D point clouds obtained using multi-view reconstruction on 3D planar city models, and registration of Kinect depth data. Naroditsky et al. (2011) showed that the external calibration of a 1D range scanner with a 2D camera results in a point-to-plane registration problem. This corresponds to the configuration of  $Points(1, 1, 1, 1, 1, 1) \leftrightarrow Planes(6)$  and hence can be seen as a special case of our point-to-plane registration. Since in their setup, all 3D points are coplanar, they have carefully derived a lower-degree solution for this problem.

<sup>1</sup><http://cmp.felk.cvut.cz/minimal/>.



**Fig. 1** The basic idea of coordinate transformation for pose estimation. It is always possible to transform the sensor coordinate system such that a chosen triplet of points  $(P_1, P_2, P_3)$  lie respectively at the

origin, on the  $X$  axis, and on the  $XY$  plane. On the other hand, the object coordinate frame can always be transformed such that  $\Pi_1$  coincides with the  $XY$  plane and  $\Pi_2$  contains the  $X$  axis

This work is an extension of our conference paper (Ramalingam et al. 2010), where we show the general idea behind point-to-plane registration and its generalization. The additional contributions in this paper in comparison to Ramalingam et al. (2010) are summarized below:

1. In Ramalingam et al. (2010), we show the derivation of point-to-plane registration algorithm for only one case where 6 points are registered to 3 planes. In this paper, we show the derivations for all cases.
2. We show the generalized registration algorithms for the following four problems:
  - Registration using three 3D point correspondences (three different point-to-plane formulations are given)
  - Mixed pose estimation using both 2D-to-3D and 3D-to-3D correspondences
  - Pose estimation using three lines (Dhome et al. 1989)
  - Pose estimation for generalized cameras (Nistér 2004)
3. We tested the above four algorithms under the generalized registration framework in both real experiments and simulations. These experiments are shown in Sect. 6.3 using data from a Kinect sensor.

### 1.2 Problem Statement

Our main goal is to compute the pose (3D translation and 3D rotation) of a sensor with respect to an object (or objects) for which a 3D model consisting of a set of planes is already known. The sensor provides the 3D coordinates of a small set of points on the object, measured in the sensor coordinate frame. We are given  $N$  points  $P_1^0, P_2^0, P_3^0, \dots, P_N^0$  from the sensor data and  $M$  planes  $\Pi_1^0, \Pi_2^0, \Pi_3^0, \dots, \Pi_M^0$  from the 3D object. We subdivide the original problem into two sub-problems:

- Compute the correspondences between the 3D points in the sensor data and the planes in the 3D object.
- Given these correspondences, compute the rotation and translation  $(R_{s2w}, T_{s2w})$  between the sensor and the object. We assume that the object lies in the world reference frame, as shown in Fig. 1.

In this paper, we explain our solution to the second problem (pose estimation given the correspondences) in Sect. 2 before discussing the correspondence problem in Sect. 4.

## 2 The Pose Estimation Problem

In this section, we develop the algorithms for pose estimation given the correspondences between the 3D points and their corresponding planes. Here we assume that the correspondences are already known—a method for computing the correspondences is explained later, in Sect. 4. We systematically consider several cases in which we know the distribution of the points on the planes (how many points correspond to each plane), developing a customized pose estimation algorithm for each case. The correspondence between a single point and a plane will yield a single coplanarity equation. Since there are 6 unknown degrees of freedom in  $(R_{s2w}, T_{s2w})$ , we need at least 6 point-to-plane correspondences to solve the pose estimation problem. There are also degenerate cases in which 6 correspondences are not sufficient. Although the individual algorithms for the various cases are slightly different, their underlying approach is the same. The algorithms for all cases are derived using the following three steps:

- *The choice of intermediate coordinate frames:* We transform the sensor and the object to intermediate coordinate frames to reduce the degree of the resulting polynomial equations. In addition, if the transformation results in a decrease in the number of degrees of freedom in the pose between the sensor and object, then the rotation  $R$  and the translation  $T$  are expressed using fewer variables.
- *The use of coplanarity constraints:* From the correspondences between the points and planes, we derive a set of coplanarity constraints. Using a linear system involving the derived coplanarity constraints, we express the unknown pose variables in a subspace spanned by one or more vectors.
- *The use of orthonormality constraints:* Finally, we use the appropriate number of orthonormality constraints from

the rotation matrix to determine solutions in the subspace just described.

### 2.1 The Choice of Intermediate Coordinate Frames

As shown in Fig. 1, we denote the original sensor frame (where the points reside) and the world reference frame (where the planes reside) by  $\mathcal{S}^0$  and  $\mathcal{W}^0$ , respectively. Our goal is to compute the transformation  $(\mathbf{R}_{\mathcal{S}2\mathcal{W}}, \mathbf{T}_{\mathcal{S}2\mathcal{W}})$  that transforms the 3D points from the sensor frame  $\mathcal{S}^0$  into the world reference frame  $\mathcal{W}^0$ . A straightforward application of coplanarity constraints in the case of 6 points would result in 6 linear equations involving 12 variables (the 9 elements of the rotation matrix  $\mathbf{R}_{\mathcal{S}2\mathcal{W}}$  and the 3 elements of the translation vector  $\mathbf{T}_{\mathcal{S}2\mathcal{W}}$ ). To solve for these variables, we would need at least 6 additional equations; these can be 6 quadratic orthonormality constraints. The solution of such a system may eventually result in a polynomial equation of degree  $64 = 2^6$ , which would have 64 solutions (upper bound as per Bezout’s theorem), and the computation of such solutions would likely be infeasible for many applications.

To overcome this difficulty, we first transform the sensor and world reference frames  $\mathcal{S}^0$  and  $\mathcal{W}^0$  to two new intermediate coordinate frames, which we call  $\mathcal{S}$  and  $\mathcal{W}$ . After this transformation, our goal is to find the remaining transformation  $(\mathbf{R}, \mathbf{T})$  between the intermediate reference frames  $\mathcal{S}$  and  $\mathcal{W}$ . We choose  $\mathcal{S}$  and  $\mathcal{W}$  so as to minimize the number of variables in  $(\mathbf{R}, \mathbf{T})$  that we need to solve for. A similar idea has been used in other problems (Dhome et al. 1989). We now define the transformations from the initial reference frames to the intermediate frames and prove that these transformations are always possible using a constructive argument.

#### 2.1.1 Transformation from $\mathcal{S}^0$ to $\mathcal{S}$

As shown in Fig. 1, we represent the  $i$ th point in  $\mathcal{S}^0$  using the notation  $P_i^0$  and the same point in  $\mathcal{S}$  using  $P_i$ . We define the transformation  $(\mathbf{R}_s, \mathbf{T}_s)$  as the one that results in the points  $(P_1, P_2, P_3)$  satisfying the following conditions: (a)  $P_1$  lies at the origin, (b)  $P_2$  lies on the positive  $\mathcal{X}$  axis, and (c)  $P_3$  lies in the  $\mathcal{XY}$  plane. Note that the points  $P_i^0$  are already given, and the transformation to the points  $P_i$  can be easily computed using the above conditions.

We denote the 3D points after the transformation as follows:

$$P_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_2 = \begin{pmatrix} X_2 \\ 0 \\ 0 \end{pmatrix}, \quad P_3 = \begin{pmatrix} X_3 \\ Y_3 \\ 0 \end{pmatrix}, \quad \text{and}$$

$$P_i = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \quad \text{for } i = \{4, 5, 6\}. \tag{1}$$

#### 2.1.2 Transformation from $\mathcal{W}^0$ to $\mathcal{W}$

We similarly represent the  $i$ th plane in  $\mathcal{W}^0$  using the notation  $\Pi_i^0$  and the same plane in  $\mathcal{W}$  using  $\Pi_i$ . We define the transformation as the one that results in the planes  $\Pi_i$  satisfying the following two conditions: (a)  $\Pi_1$  coincides with the  $\mathcal{XY}$  plane, and (b)  $\Pi_2$  contains the  $\mathcal{X}$  axis.

Assume that  $Q_1^0$  and  $Q_2^0$  are two points on the line of intersection of the two planes  $\Pi_1^0$  and  $\Pi_2^0$ . Let  $Q_3^0$  be any other point on the plane  $\Pi_1^0$ . Let  $Q_1, Q_2,$  and  $Q_3$  denote the same 3D points after the transformation from  $\mathcal{W}^0$  to  $\mathcal{W}$ . The required transformation  $(\mathbf{R}_w, \mathbf{T}_w)$  is the one that maps the triplet  $(Q_1^0, Q_2^0, Q_3^0)$  to  $(Q_1, Q_2, Q_3)$ . Note that three points  $Q_i^0$  satisfying the description above can be easily determined from the planes  $\Pi_i^0$ , and the transformation from points  $Q_i^0$  to points  $Q_i$  can be computed in the same way as the transformation described above from points  $P_i^0$  to points  $P_i$ .

We write the equations of the planes after the transformation as follows:

$$Z = 0 : \Pi_1 \tag{2}$$

$$B_2Y + C_2Z = 0 : \Pi_2 \tag{3}$$

$$A_iX + B_iY + C_iZ + D_i = 0 : \Pi_i, \quad \text{for } i = \{3, 4, 5, 6\}. \tag{4}$$

#### 2.1.3 Point-to-Plane Assignment

Depending on the particular configuration  $Points(a_1, \dots, a_n) \leftrightarrow Planes(n)$  of the points and planes, we choose which sensor points correspond to each of  $P_1, P_2, \dots$ , and which object planes correspond to each of  $\Pi_1, \Pi_2, \dots$ , so as to minimize the number of variables in the transformation between the intermediate frames.

In the remainder of this subsection, and in the following Sects. 2.2 and 2.3, we explain the method in the context of a particular example: the configuration  $Points(3, 2, 1) \leftrightarrow Planes(3)$ . For this configuration, we may without loss of generality assume the following correspondences between the points and the planes:

$$\Pi_1 \leftarrow \{P_1, P_2, P_3\}, \quad \Pi_2 \leftarrow \{P_4, P_5\},$$

$$\Pi_3 \leftarrow \{P_6\}. \tag{5}$$

As a result of this assignment, the plane corresponding to the three points  $\{P_1, P_2, P_3\}$  and the plane  $\Pi_1$  are both mapped to the  $\mathcal{XY}$  plane. The final rotation  $(\mathbf{R})$  and translation  $(\mathbf{T})$  between the intermediate sensor coordinate frame  $\mathcal{S}$  and the intermediate object coordinate frame  $\mathcal{W}$  must preserve the coplanarity of these three points and their corresponding plane. Thus, the final transformation can be chosen so as to map all points on the  $\mathcal{XY}$  plane to points on the  $\mathcal{XY}$  plane.

In other words, the rotation should be only along the  $Z$  axis and the translation along the  $X$  and  $Y$  axes. There are two pairs of rotation and translation that satisfy this constraint:

$$\begin{aligned}
 \mathbf{R}_1 &= \begin{pmatrix} R_{11} & R_{12} & 0 \\ -R_{12} & R_{11} & 0 \\ 0 & 0 & 1 \end{pmatrix}, & \mathbf{T}_1 &= \begin{pmatrix} T_1 \\ T_2 \\ 0 \end{pmatrix}; \\
 \mathbf{R}_2 &= \begin{pmatrix} R_{11} & R_{12} & 0 \\ R_{12} & -R_{11} & 0 \\ 0 & 0 & -1 \end{pmatrix}, & \mathbf{T}_2 &= \begin{pmatrix} T_1 \\ T_2 \\ 0 \end{pmatrix}
 \end{aligned} \tag{6}$$

By choosing assignment (5) and separately formulating  $\mathbf{R}_1$  and  $\mathbf{R}_2$ , we have minimized the number of degrees of freedom to solve for in the transformation between the intermediate frames of reference. Note that  $\mathbf{R}_1$  and  $\mathbf{R}_2$  are related to each other by a  $180^\circ$  rotation about the  $X$  axis. Below, we explain the algorithm for solving for  $\mathbf{R}_1$  and  $\mathbf{T}_1$ .

### 2.2 The Use of Coplanarity Constraints

To explain our method’s use of coplanarity constraints (and orthonormality constraints), we continue with the example of the specific configuration  $Points(3, 2, 1) \leftrightarrow Planes(3)$ . We know that the points  $P_4$  and  $P_5$  lie on the plane  $\Pi_2$ , whose equation is given by (3). This implies that these points must satisfy the following coplanarity constraints:

$$B_2(-R_{12}X_i + R_{11}Y_i + T_2) + C_2Z_i = 0, \quad \text{for } i = \{4, 5\} \tag{7}$$

Similarly, the constraint from the third plane  $\Pi_3$  is given below:

$$\begin{aligned}
 A_3(R_{11}X_6 + R_{12}Y_6 + T_1) + B_3(-R_{12}X_6 + R_{11}Y_6 + T_2) \\
 + C_3Z_6 + D_3 = 0
 \end{aligned} \tag{8}$$

Using the coplanarity constraints (7), (8), we construct the following linear system:

$$\begin{aligned}
 \underbrace{\begin{pmatrix} B_2Y_4 & -B_2X_4 & 0 & B_2 \\ B_2Y_5 & -B_2X_5 & 0 & B_2 \\ A_3X_6 + B_3Y_6 & A_3Y_6 - B_3X_6 & A_3 & B_3 \end{pmatrix}}_{\mathcal{A}} \begin{pmatrix} R_{11} \\ R_{12} \\ T_1 \\ T_2 \end{pmatrix} \\
 = \begin{pmatrix} -C_2Z_4 \\ -C_2Z_5 \\ -C_3Z_6 - D_3 \end{pmatrix}
 \end{aligned} \tag{9}$$

The matrix  $\mathcal{A}$  consists of known values and has rank 3. As there are 4 variables in the linear system, we can obtain their

solution in a subspace spanned by one vector:

$$\begin{pmatrix} R_{11} \\ R_{12} \\ T_1 \\ T_2 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} + l_1 \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}, \tag{10}$$

where the values  $u_i, v_i$  are known, and  $l_1$  is the only unknown variable.

### 2.3 The Use of Orthonormality Constraints

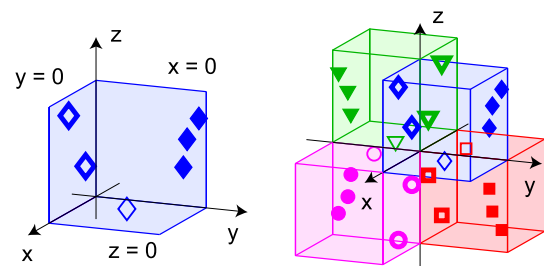
We can solve for the unknown variable  $l_1$  using a single orthonormality constraint ( $R_{11}^2 + R_{12}^2 = 1$ ) for the rotation variables.

$$(u_1 + l_1v_1)^2 + (u_2 + l_1v_2)^2 = 1 \tag{11}$$

By solving the above equation, we obtain two different solutions for  $l_1$ . As a result, we obtain two solutions for the transformation  $(\mathbf{R}_1, \mathbf{T}_1)$ . Since we can similarly compute two solutions for  $(\mathbf{R}_2, \mathbf{T}_2)$ , we finally have 4 solutions for  $(\mathbf{R}, \mathbf{T})$ . Using the obtained solutions for  $(\mathbf{R}, \mathbf{T})$ , the transformation between the original coordinate frames  $(\mathbf{R}_{s2w}, \mathbf{T}_{s2w})$  can be easily computed.

### 2.4 Visualization of Solutions

There is a geometric relationship between the multiple solutions obtained for the transformation  $(\mathbf{R}, \mathbf{T})$ . For example, in Fig. 2, we show the four solutions derived above, for a special case in which the 3 planes are orthogonal to each other. All of the solutions satisfy the same set of plane equations, but they exist in different octants. Every solution is just a rotation of another solution about one of the three axes by  $180^\circ$ . If we slightly modify the planes so that they are no longer orthogonal, the different solutions start to drift away from each other.



**Fig. 2** Left: Original setting. Right: Visualization of 4 solutions (each denoted by diamonds, squares, circles, and triangles) for the points lying on the 3 orthogonal planes. The blue solution corresponds to the original setting while the other 3 still satisfy the coplanarity and orthonormality constraints (Color figure online)

**Table 1** Point-to-plane configurations and their solutions. Each row of the table presents a different configuration, in which  $n$  denotes the number of distinct planes and each  $a_i$  refers to the number of points that lie on the  $i$ th plane. The first two rows show the degenerate cases for which there is an insufficient number of points or planes. The next four rows consider non-minimal solutions using more than 6 points. The remaining rows show several minimal configurations (each us-

ing exactly 6 points). The number of solutions is given, followed by the average number of real (non-imaginary) solutions in parentheses based on 1000 computations from the simulation described in Sect. 6. Processing time was measured using a MATLAB implementation on a 2.66 GHz PC; the symbol † indicates the use of Groebner basis methods (Kukelova et al. 2008)

$n$	$(a_1, \dots, a_n)$	Assignment	# of Solutions	Process time (ms)
$< 3$	–	–	degenerate	–
$n$	$\sum a_i < 6$	–	degenerate	–
3	(3, 3, 3)	$\Pi_1 \leftarrow \{P_1, P_2, P_3\}$ $\Pi_2 \leftarrow \{P_4, P_5, P_6\}$ $\Pi_3 \leftarrow \{P_7, P_8, P_9\}$	2 (2)	5
3	(3, 3, 2)	$\Pi_1 \leftarrow \{P_1, P_2, P_3\}$ $\Pi_2 \leftarrow \{P_4, P_5, P_6\}$ $\Pi_3 \leftarrow \{P_7, P_8\}$	2 (2)	5
3	(3, 3, 1)	$\Pi_1 \leftarrow \{P_1, P_2, P_3\}$ $\Pi_2 \leftarrow \{P_4, P_5, P_6\}$ $\Pi_3 \leftarrow \{P_7\}$	2 (2)	5
3	(3, 2, 2)	$\Pi_1 \leftarrow \{P_1, P_2, P_3\}$ $\Pi_2 \leftarrow \{P_4, P_5\}$ $\Pi_3 \leftarrow \{P_6, P_7\}$	2 (2)	5
3	(4, 1, 1)	–	degenerate	–
3	(3, 2, 1)	$\Pi_1 \leftarrow \{P_1, P_2, P_3\}$ $\Pi_2 \leftarrow \{P_4, P_5\}$ $\Pi_3 \leftarrow \{P_6\}$	4 (4)	6
3	(2, 2, 2)	$\Pi_1 \leftarrow \{P_5, P_6\}$ $\Pi_2 \leftarrow \{P_3, P_4\}$ $\Pi_3 \leftarrow \{P_1, P_2\}$	8 (4.4)	140 <sup>†</sup>
4	(3, 1, 1, 1)	$\Pi_1 \leftarrow \{P_1, P_2, P_3\}$ $\Pi_2 \leftarrow \{P_4\}$ $\Pi_3 \leftarrow \{P_5\}$ $\Pi_4 \leftarrow \{P_6\}$	4 (2.8)	6
4	(2, 2, 1, 1)	$\Pi_1 \leftarrow \{P_5, P_6\}$ $\Pi_2 \leftarrow \{P_3, P_4\}$ $\Pi_3 \leftarrow \{P_2\}$ $\Pi_4 \leftarrow \{P_1\}$	8 (3.6)	140 <sup>†</sup>
5	(2, 1, 1, 1, 1)	$\Pi_1 \leftarrow \{P_5, P_6\}, \Pi_i \leftarrow \{P_{6-i}\}, i = \{3, 4, 5\}$	16 (5.8)	410 <sup>†</sup>
6	(1, 1, 1, 1, 1, 1)	$\Pi_i \leftarrow \{P_{6-i+1}\}, i = \{1, 2, 3, 4, 5, 6\}$	16 (5.8)	1200 <sup>†</sup>

### 3 Other Configurations

The example shown above is one of the easiest point-to-plane registration algorithms to derive. Several harder configurations also arise from the distribution of 6 (or more) distinct points on 3 or more planes (see Table 1). We have solved every case using the same intermediate transformation technique described above. All of the different scenarios, the corresponding assignments of points and planes, and the number of solutions are summarized in Table 1.

The key to solving each configuration is to determine a point-to-plane assignment that minimizes the number of variables appearing in the transformation  $(\mathbf{R}, \mathbf{T})$  between the intermediate frames. In general, such an optimal assignment can be found by considering different point-to-plane assignments and checking the resulting coplanarity constraint equations for the 6 points and their corresponding planes. For example, in the configuration  $Points(3, 2, 1) \leftrightarrow Planes(3)$ , the point-to-plane assignments given in (5) minimize the number of unknowns in Eq. (6) for  $(\mathbf{R}, \mathbf{T})$ .



We show that the derivation is essentially the same as the  $Points(3, 2, 1) \leftrightarrow Planes(3)$  configuration for the following cases:

- $Points(3, 3, 3) \leftrightarrow Planes(3)$
- $Points(3, 3, 2) \leftrightarrow Planes(3)$
- $Points(3, 3, 1) \leftrightarrow Planes(3)$
- $Points(3, 2, 2) \leftrightarrow Planes(3)$
- $Points(3, 1, 1, 1) \leftrightarrow Planes(4)$

Here we present a constructive example for the  $Points(3, 1, 1, 1) \leftrightarrow Planes(4)$  configuration. We also present the  $Points(2, 2, 2) \leftrightarrow Planes(3)$  configuration, which is less trivial and a representative case of the other configurations.

### 3.1 $Points(3, 1, 1, 1) \leftrightarrow Planes(4)$

First we perform the intermediate transformation as shown in Sect. 2.1. We denote the 3D points  $P_i, i = \{1, \dots, 6\}$  using Eq. (1) and the planes  $\Pi_j, j = \{1, \dots, 6\}$  using Eqs. (2), (3) and (4). In this configuration, we use the following assignment:

$$\begin{aligned} \Pi_1 &\Leftarrow \{P_1, P_2, P_3\}, & \Pi_2 &\Leftarrow \{P_4\}, \\ \Pi_3 &\Leftarrow \{P_5\}, & \Pi_4 &\Leftarrow \{P_6\} \end{aligned} \tag{12}$$

Same as the  $Points(3, 2, 1) \leftrightarrow Planes(3)$  configuration, this assignment enables to restrict the rotation between the intermediate coordinate frames only along the  $Z$  axis, and the translation along the  $X$  and  $Y$  axes. The rotation and translation can therefore be described as in Eq. (6). For the first set of  $(R_1, T_1)$ , we obtain the following three equations from the coplanarity constraints obtained from points lying on  $\Pi_2, \Pi_3$ , and  $\Pi_4$ :

$$\underbrace{\begin{pmatrix} B_2Y_4 & -B_2X_4 & 0 & B_2 \\ A_3X_5 + B_3Y_5 & A_3Y_5 - B_3X_5 & A_3 & B_3 \\ A_4X_6 + B_4Y_6 & A_4Y_6 - B_4X_6 & A_4 & B_4 \end{pmatrix}}_A \begin{pmatrix} R_{11} \\ R_{12} \\ T_1 \\ T_2 \end{pmatrix} = \begin{pmatrix} -C_2Z_4 \\ -C_3Z_5 - D_3 \\ -C_4Z_6 - D_4 \end{pmatrix} \tag{13}$$

Since the rank of  $\mathcal{A}$  is 3, we obtain the solution of the variables in a subspace shown in Eq. (10). We obtain two solutions for  $l_1$  using an orthonormality constraint  $R_{11}^2 + R_{12}^2 = 1$  and thereby two solutions for  $(R_1, T_1)$ . Similarly, we obtain two solutions for the second set  $(R_2, T_2)$ . As a result, we obtain a total of 4 solutions for this configuration.

*Similar Configurations* As described above, the (non-minimal) configurations containing three points on a single plane can be solved similarly. We assign the three points  $\{P_1, P_2, P_3\}$  to the plane  $\Pi_1$ , which enables to formulate the rotation and translation variables as in (6). Since these configurations have more than 6 points (7, 8, and 9), we include additional coplanarity constraints in (13). This makes the linear system (13) over-determined and a single solution can be obtained as the least-squares solution. Since the solution can be obtained for  $(R_1, T_1)$  and  $(R_2, T_2)$  separately, we finally obtain 2 solutions.

### 3.2 $Points(2, 2, 2) \leftrightarrow Planes(3)$

In this configuration, the assignment

$$\begin{aligned} \Pi_1 &\Leftarrow \{P_5, P_6\}, & \Pi_2 &\Leftarrow \{P_3, P_4\}, \\ \Pi_3 &\Leftarrow \{P_1, P_2\} \end{aligned} \tag{14}$$

minimizes the number of variables as

$$R = \begin{pmatrix} R_{11} & * & * \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}, \quad T = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \end{pmatrix}, \tag{15}$$

where  $*$  indicates terms that are multiplied by zero in the coplanarity constraints (these values will be determined later from the other 7 values in this rotation matrix).

Using the above 10 variables and 6 coplanarity constraints obtained from the 6 points and their corresponding planes, we construct the following system, in which  $\mathcal{A}$  is a  $6 \times 10$  matrix:

$$\underbrace{\begin{pmatrix} 0 & 0 & 0 & 0 & X_5 & Y_5 & Z_5 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & X_6 & Y_6 & Z_6 & 0 & 0 & 1 \\ 0 & B_2X_3 & B_2Y_3 & 0 & C_2X_3 & C_2Y_3 & 0 & 0 & B_2 & C_2 \\ 0 & B_2X_4 & B_2Y_4 & B_2Z_4 & C_2X_4 & C_2Y_4 & C_2Z_4 & 0 & B_2 & C_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & A_3 & B_3 & C_3 \\ A_3X_2 & B_3X_2 & 0 & 0 & C_3X_2 & 0 & 0 & A_3 & B_3 & C_3 \end{pmatrix}}_A \begin{pmatrix} R_{11} \\ R_{21} \\ R_{22} \\ R_{23} \\ R_{31} \\ R_{32} \\ R_{33} \\ T_1 \\ T_2 \\ T_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -D_3 \\ -D_3 \end{pmatrix} \tag{16}$$

The solution of this equation can be obtained in a subspace of dimension 4, resulting in 4 unknown parameters ( $l_1, l_2, l_3, l_4$ ).

$$\begin{pmatrix} R_{11} \\ R_{21} \\ R_{22} \\ R_{23} \\ R_{31} \\ R_{32} \\ R_{33} \\ T_1 \\ T_2 \\ T_3 \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \end{pmatrix} + l_1 \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \\ b_{10} \end{pmatrix} + l_2 \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \\ c_{10} \end{pmatrix} + l_3 \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \\ d_9 \\ d_{10} \end{pmatrix} + l_4 \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \\ e_{10} \end{pmatrix} \tag{17}$$

To solve for the unknown parameters, we use 4 orthonormality constraints for the rotation variables:

$$R_{11}^2 + R_{21}^2 + R_{31}^2 = 1 \tag{18}$$

$$R_{21}^2 + R_{22}^2 + R_{23}^2 = 1 \tag{19}$$

$$R_{31}^2 + R_{32}^2 + R_{33}^2 = 1 \tag{20}$$

$$R_{21}R_{31} + R_{22}R_{32} + R_{23}R_{33} = 0 \tag{21}$$

By substituting all the rotation variables (17) including only four unknowns ( $l_1, l_2, l_3, l_4$ ) to (18)–(21), we obtain a final set of equations (4 second-order equations). To efficiently solve them, we used a software using Groebner basis method (Kukelova et al. 2008). The solver finally provided 8 solutions from the set of equations in this configuration.

*Similar Configurations* The following configurations can be solved similarly (the corresponding point-to-plane assignments are shown in Table 1):

- *Points*(2, 2, 1, 1) ↔ *Planes*(4)

The rotation and translation variables which appear in the coplanarity constraints are the same as in (15). The linear system (16) also has the same number of equations. We therefore obtain 8 solutions.

- *Points*(2, 1, 1, 1, 1) ↔ *Planes*(5)

The rotation and translation variables are formulated as

$$\mathbf{R} = \begin{pmatrix} R_{11} & R_{12} & * \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}, \quad \mathbf{T} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \end{pmatrix}, \tag{22}$$

which include 11 variables. The linear system (16) is constructed with a  $6 \times 11$  matrix  $\mathcal{A}$  whose rank is 6, so the solution is obtained in a subspace of dimension 5 with 5 unknown parameters. These parameters are solved using orthonormality constraints for the rotation variables. We have the following additional orthonormality constraint in addition to the four Eqs. (18), (19), (20) and (21).

$$R_{11}R_{12} + R_{21}R_{22} + R_{31}R_{32} = 0 \tag{23}$$

Finally 16 solutions are obtained from the set of equations.

- *Points*(1, 1, 1, 1, 1, 1) ↔ *Planes*(6)

In this case, we need to solve for all 12 variables of rotation and translation. The linear system (16) is constructed with a  $6 \times 12$  matrix  $\mathcal{A}$  and the solution is obtained in a subspace of dimension 6 with 6 unknown parameters. Using six orthonormality constraints for the rotation variables, the solver using Groebner basis method (Kukelova et al. 2008) provides 16 solutions. In addition to the 5 orthonormality constraints used for solving the case *Points*(2, 1, 1, 1, 1) ↔ *Planes*(5), we use the following additional orthonormality constraint:

$$R_{12}R_{13} + R_{22}R_{23} + R_{32}R_{33} = 0 \tag{24}$$

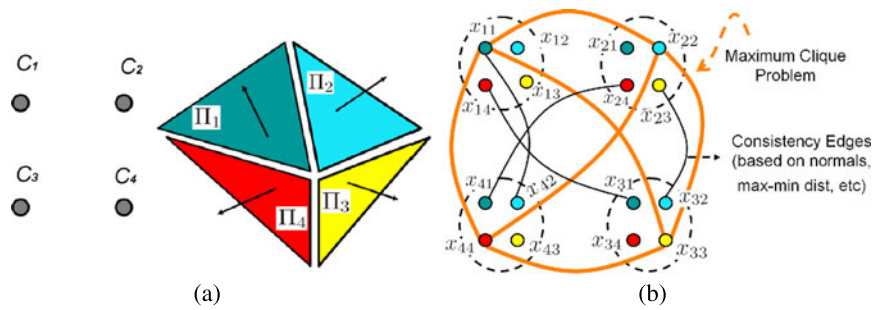
As shown in Table 1, this requires the longest processing time, which clarifies the advantage of using our intermediate transformation to reduce the variables to be solved in the final set of equations.

### 3.3 Degenerate Cases

Table 1 includes several degenerate cases based on the number of points and planes. In addition, degeneracies can occur based on the geometry of the planes. In the case of 3 planes, if the  $3 \times 3$  matrix consisting of all three normals has rank less than 3 (e.g., if two of the three planes are parallel), it is a degenerate configuration.

## 4 The Correspondence Problem

In the previous section, we assumed that the point-to-plane correspondences were known. In this section, we briefly de-



**Fig. 3** (a) The problem of finding correspondences between clusters of points  $C_i$  and planes  $\Pi_j$ . (b) This can be formulated as a maximum clique problem. Each node  $x_{ij}$  in this graph represents a mapping between cluster  $C_i$  and plane  $\Pi_j$ . An edge between two nodes is a

consistency edge, signifying that both of these mappings can occur simultaneously without conflicting with the three constraints given in Grimson and Lozano-Pérez (1983)

scribe a method to compute these correspondences. The basic idea of the correspondence problem and the geometrical constraints involved in identifying feasible correspondences are explained in detail in Grimson and Lozano-Pérez (1983) using an interpretation tree approach. The same problem can also be formulated as graph-theoretical problems such as independent set, vertex cover and maximum clique (Grimson and Lozano-Pérez 1983; Enqvist et al. 2009; Tu et al. 1999).

Our goal in this section is to compute all of the feasible mappings (possible assignments) between the 3D points in the sensor domain and planes in the object. Feasible mappings refer to correspondences that satisfy the many geometrical constraints arising from the angles between the normals, pairwise distances, etc. (Grimson and Lozano-Pérez 1983). Although such constraints do not always guarantee the correctness of the mappings, a wrong correspondence seldom exists satisfying all the constraints. In addition, since we use them in hypothesize-and-test algorithms such as RANSAC, outliers can be detected and removed.

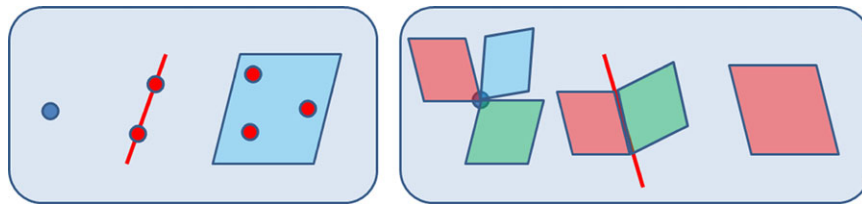
In what follows, we briefly explain our approach using the maximum clique problem formulation. First, we cluster the points from the sensor into several planes, denoting the  $i$ th cluster as  $C_i$ . Note that each cluster may contain multiple points or even just a single point. As shown in Fig. 3(a), our goal is to map these clusters to the corresponding planes  $\Pi_j$  in the object. In order to do this, we construct a graph as shown in Fig. 3(b). Every node in this graph  $x_{ij}$  represents a mapping between the cluster  $C_i$  (from the sensor) and the plane  $\Pi_j$  (from the object). An edge between  $x_{ij}$  and  $x_{kl}$  is referred to as a consistency edge that signifies that both these mappings can occur simultaneously without conflicting with the three constraints given in Grimson and Lozano-Pérez (1983). The feasible correspondences between points and planes can be obtained by finding the maximum clique in the graph. A maximum clique for a graph refers to the largest subset of nodes in which each pair of nodes in the subset is connected by an edge. In the graph we constructed,

finding a maximum clique provides us a set of mappings in which all possible pairwise consistencies are satisfied.

Several techniques can be used to solve these NP-hard problems (Enqvist et al. 2009; Li and Hartley 2007). Since we use minimal approaches for our applications, we are not interested in the correspondences for all of the points in the registration problem. Instead, we are concerned with identifying a small number of point-to-plane correspondences (sufficient to resolve issues from degeneracies and outliers). In fact, one of the main advantages of the proposed minimal solution is that it only requires correspondences for a small number of points. This enabled us to use a simple tree-based search for finding the maximum cliques in the real-world experiments described in Sect. 6.

## 5 A General Framework for Pose Estimation

We extend our solution for the point-to-plane problem to a unified pose estimation framework for most 2D-to-3D and 3D-to-3D registrations. Several 2D-to-3D pose estimation algorithms have been proposed in the literature (Horn 1987; Dhome et al. 1989; Chen and Medioni 1991; Olsson et al. 2006; Haralick et al. 1994; Grimson and Lozano-Pérez 1983; Chen 1991; Nistér 2004). All these pose estimation algorithms involve the registration of one set of geometrical entities (points, lines, or planes) to another. For example, in the case of generalized pose estimation, we register three 3D points to the corresponding non-parametric projection rays from the cameras to compute the pose of the object with respect to the camera (Nistér 2004). In the case of 2D-to-3D pose estimation using three lines, we can look at this problem as a registration of three interpretation planes (each formed by two projection rays corresponding to a single line in the image) on three lines (Dhome et al. 1989). In the case of 3D-to-3D line-to-plane registration, we register lines from the sensor data to planes from the object (Chen 1991). In the case of 3D-to-3D point-to-point registration,



**Fig. 4** A general framework to transform a given registration problem to a point-to-plane problem. *Left:* In the sensor data, we transform all geometrical entities (*points, lines and planes*) to points. A point is preserved as a point. In the case of lines and planes we sample two and

three arbitrary points, respectively. *Right:* In the object data, we convert all geometrical entities to planes. A plane is preserved as a plane. Points and lines are parameterized using 3-plane and 2-plane representations, as shown

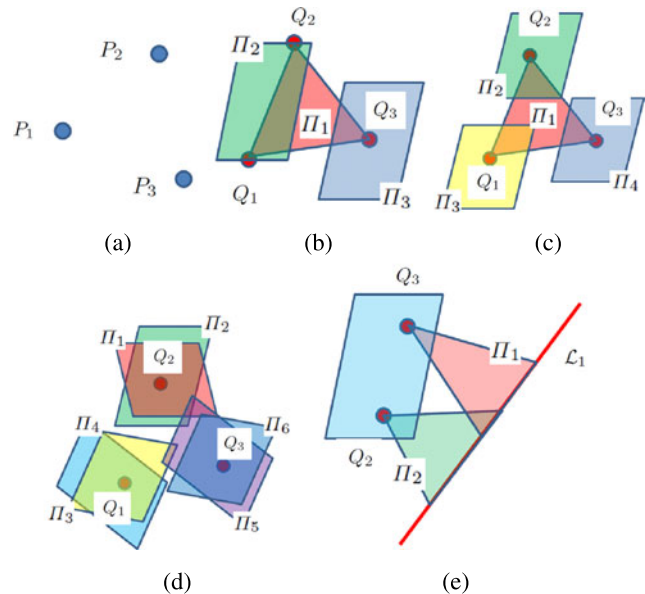
we register points from sensor data to points in the object (Horn 1987). One could also propose registration algorithms involving mixture of geometrical entities and thereby we could have more than 20 2D-to-3D and 3D-to-3D registration scenarios. We emphasize that any of these pose estimation algorithms involving any combination of geometrical entities to any other combination could be transformed to a point-to-plane registration algorithm and solved using the following simple algorithm.

1. In the sensor data, we transform all the geometrical entities (points, lines and planes) to points. This is done using 2-point and 3-point representation of lines and planes respectively as shown in Fig. 4.
2. In the object data, we transform all the geometrical entities to planes. This is done by 3-plane and 2-plane representations for points and lines, respectively. Note that the 3 planes passing through a point need not be orthogonal. Similarly, we can also use 2 non-orthogonal planes to represent a line. The appropriate choice of these planes plays a crucial role in obtaining an efficient pose estimation algorithm.
3. After these transformations, we can use our point-to-plane registration algorithm.

In the remaining part of this section, we show a few examples of registration problems that can be solved using the theory developed for point-to-plane registration.

### 5.1 Registration using Three 3D Point Correspondences

Given three correspondences between 3D points in one reference frame to 3D points in another reference frame, the goal is find a transformation to register the points. Let  $P_1, P_2$  and  $P_3$  correspond to three 3D points in the first reference frame. Let the corresponding 3D points in the second reference frame be given by  $Q_1, Q_2$  and  $Q_3$  respectively. In order to register, we first parameterize the three points in the second reference frame using planes. We show three different ways of doing it as shown in Figs. 5(b), (c) and (d). In Figs. 5(b) we consider three planes  $\Pi_1, \Pi_2$  and  $\Pi_3$  satisfying the following conditions:

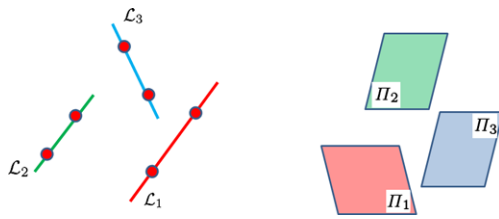


**Fig. 5** We show the transformation of geometrical entities to map several registration problems to a point-to-plane problem. (a) Let us consider three 3D points  $P_1, P_2$  and  $P_3$  in the first reference frame. (b), (c) and (d) show the corresponding 3D points in the second reference frame with different plane parameterizations. In (e) we show a mixed registration scenario where the three points  $P_1, P_2$  and  $P_3$  correspond to one 3D line  $\mathcal{L}_1$  and two 3D points  $Q_2$  and  $Q_3$  respectively

- $\Pi_1$  contains all three points  $Q_1, Q_2$  and  $Q_3$ .
- $\Pi_2$  contains two points  $Q_1$  and  $Q_3$ .
- $\Pi_3$  contains only one point  $Q_3$ .

It is easy to construct the plane  $\Pi_1$  passing through three points. In the case of  $\Pi_2$  and  $\Pi_3$ , we chose one or two additional random points to compute the plane parameters. The registration problem is now equivalent to the point-to-plane algorithm corresponding to the configuration  $Points(3, 2, 1) \leftrightarrow Planes(3)$  where the following constraints should be satisfied after the registration:

- $\Pi_1$  must contain all three points  $P_1, P_2$  and  $P_3$ .
- $\Pi_2$  must contain two points  $P_1$  and  $P_3$ .
- $\Pi_3$  must contain only one point  $P_3$ .



**Fig. 6** The registration of three lines to three planes is shown. Let  $\mathcal{L}_1$ ,  $\mathcal{L}_2$  and  $\mathcal{L}_3$  be three 3D lines and the corresponding planes be given by  $\Pi_1$ ,  $\Pi_2$  and  $\Pi_3$  respectively. We randomly sample two points on each line and transform the given problem to a point-to-plane registration problem  $Points(2, 2, 2) \leftrightarrow Planes(3)$

Similarly, Figs. 5(c) and (d) show the transformations to the following configurations:

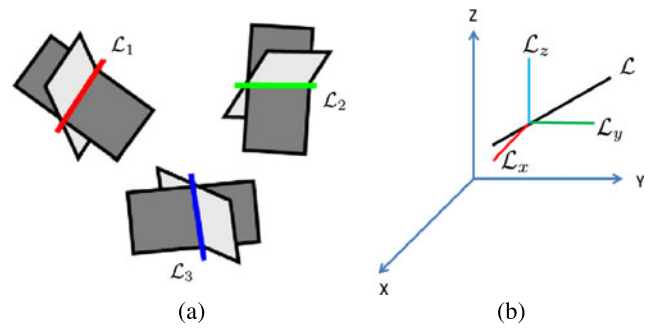
- $Points(3, 1, 1, 1) \leftrightarrow Planes(4)$
- $Points(1, 1, 1, 1, 1, 1) \leftrightarrow Planes(6)$

In experiments we show that the configuration  $Points(3, 2, 1) \leftrightarrow Planes(3)$  is marginally better than the others.

### 5.2 Line-to-Plane Registration

Let us consider the 3D-to-3D line to plane registration algorithm (Chen 1991) and 2D-to-3D pose estimation problem using three lines (Dhome et al. 1989). In Chen (1991) we are given three 3D lines from the sensor data and three 3D planes from the object. Our goal is to compute the pose  $(\mathbf{R}, \mathbf{T})$  such that the three lines register on their corresponding three planes. In Dhome et al. (1989) we are given three 3D lines in the world coordinate frame and their corresponding three 2D lines in the image space. Our goal is to compute the pose  $(\mathbf{R}, \mathbf{T})$  that registers the 3D lines from the world to the camera reference frame. In this problem, the rays corresponding to imaged pixels of a single 3D line in space lie on a plane. This plane is referred to as interpretation plane in Dhome et al. (1989). The 2D-to-3D pose estimation using three lines could thus be seen as a registration of three 3D lines to three planes. Hence, both these problems (Dhome et al. 1989; Chen 1991) can be seen as the registration of lines to planes.

We transform the geometrical entities in one coordinate frame to points and the other to planes. Here we sample two points on every line and thus obtain 6 points from three lines. In the other coordinate frame we already have three planes. This problem can be seen as the case  $Points(2, 2, 2) \leftrightarrow Planes(3)$  of the point-to-plane registration algorithm as shown in Fig. 6. A total of 8 solutions is obtained for this problem, which agrees with the number of solutions in Dhome et al. (1989) and Chen (1991).



**Fig. 7** (a) The registration of three points to three lines: this problem is transformed to the point-to-plane registration problem  $Points(1, 1, 1, 1, 1, 1) \leftrightarrow Planes(6)$ . (b) A constructive approach to show that it is possible to choose three planes  $\Pi_x, \Pi_y, \Pi_z$  along every line  $\mathcal{L}$  that are parallel to the three orthogonal coordinate axes. We chose a point on the line  $\mathcal{L}$  and consider orthogonal lines  $\mathcal{L}_x, \mathcal{L}_y, \mathcal{L}_z$ , passing through this point and parallel to the three axes. For every  $i = \{x, y, z\}$  the plane  $\Pi_i$  is given by the one that passes through the pair of lines  $(\mathcal{L}, \mathcal{L}_i)$

### 5.3 Point-to-Line Registration

Consider the 3D-to-3D registration of three points  $P_1, P_2$  and  $P_3$  on their corresponding lines  $\mathcal{L}_1, \mathcal{L}_2$  and  $\mathcal{L}_3$  respectively. The underlying registration problem in the case of generalized pose estimation is essentially same as (Nistér 2004). In Nistér (2004) the goal is to register three 3D points to their corresponding projection rays in a non-parametric imaging model.

Using our intermediate transformation technique, we have the following:

$$P_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_2 = \begin{pmatrix} X_2 \\ 0 \\ 0 \end{pmatrix}, \quad P_3 = \begin{pmatrix} X_3 \\ Y_3 \\ 0 \end{pmatrix} \quad (25)$$

For every line we know a point  $\mathbf{A}_i$  on the line and its direction  $\mathbf{d}_i$ :

$$\mathbf{A}_i = \begin{pmatrix} A_{ix} \\ A_{iy} \\ A_{iz} \end{pmatrix}, \quad \mathbf{d}_i = \begin{pmatrix} a_i \\ b_i \\ c_i \end{pmatrix}, \quad i = \{1, 2, 3\} \quad (26)$$

Our goal is to transform the given point-to-line registration to a point-to-plane one. In order to do this, we use 2-plane parameterization for every line  $\mathcal{L}$  such that the two planes intersect uniquely at  $\mathcal{L}$ . Without loss of generality, we chose the following two planes for  $\mathcal{L}_1$ :

$$b_1X - a_1Y + D_1 = 0 : \Pi_1 \quad (27)$$

$$c_1X - a_1Z + D_2 = 0 : \Pi_2 \quad (28)$$

Using the point  $\mathbf{A}$  on the line  $\mathcal{L}$  we can compute  $D_1$  and  $D_2$ . Figure 7(b) shows the geometrical configuration of these planes. The planes  $\Pi_1$  and  $\Pi_2$  are parallel to  $Z$  and  $Y$  axes

respectively. It is possible to construct three planes  $\Pi_x, \Pi_y$  and  $\Pi_z$  passing through every line  $\mathcal{L}$  in 3D space. In order to do this, we first consider an arbitrary point  $P$  on the line  $\mathcal{L}$ . Let  $\mathcal{L}_x, \mathcal{L}_y$  and  $\mathcal{L}_z$  be three orthogonal lines passing through the point  $P$ . For every  $i = \{x, y, z\}$ , the plane  $\Pi_i$  is formed by the pair of intersecting lines  $(\mathcal{L}, \mathcal{L}_i)$ . Note that a line  $\mathcal{L}$  lies on a plane  $\Pi$  only if its direction vector  $\mathbf{d}$  and the plane's normal  $\mathbf{n}$  satisfy the condition  $\mathbf{d}^T \mathbf{n} = 0$ . Accordingly the normals of the planes  $\Pi_x, \Pi_y$  and  $\Pi_z$  are given by  $\mathbf{n}_x = (0, c_1, -b_1)$ ,  $\mathbf{n}_y = (c_1, 0, -a_1)$  and  $\mathbf{n}_z = (b_1, -a_1, 0)$  respectively. It is important to observe that the planes  $\Pi_x, \Pi_y$  and  $\Pi_z$  are not orthogonal to each other.

Note that the normal vectors of the above two planes are orthogonal to the direction vector of  $\mathcal{L}_1$ . Similarly we chose the following planes for lines  $\mathcal{L}_2(\Pi_3, \Pi_4)$  and  $\mathcal{L}_3(\Pi_5, \Pi_6)$ :

$$b_2X - a_2Y + D_3 = 0 : \Pi_3 \tag{29}$$

$$c_2X - a_2Z + D_4 = 0 : \Pi_4 \tag{30}$$

$$b_3X - a_3Y + D_5 = 0 : \Pi_5 \tag{31}$$

$$c_3X - a_3Z + D_6 = 0 : \Pi_6 \tag{32}$$

Now we have 6 point-to-plane correspondences:

$$\begin{aligned} \Pi_i &\Leftarrow \{P_1\}, & i &= \{1, 2\} \\ \Pi_j &\Leftarrow \{P_2\}, & j &= \{3, 4\} \\ \Pi_k &\Leftarrow \{P_3\}, & k &= \{5, 6\} \end{aligned} \tag{33}$$

The above mapping corresponds to the case  $Points(1, 1, 1, 1, 1, 1) \leftrightarrow Planes(6)$ , which yields 16 different solutions. However in this specific case, we show that it is possible to solve the problem efficiently using an 8th degree polynomial, yielding only 8 solutions. Due to the appropriate choice of the planes, we minimize the number of variables as

$$\mathbf{R} = \begin{pmatrix} R_{11} & R_{12} & * \\ R_{21} & R_{22} & * \\ R_{31} & R_{32} & * \end{pmatrix}, \quad \mathbf{T} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \end{pmatrix}, \tag{34}$$

where \* indicates terms that are multiplied by zero in the coplanarity constraints (these values will be determined later from the other 6 values in this rotation matrix). The coplanarity constraints lead to the following system with 9 variables and 6 equations:

$$\underbrace{\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & b_1 & -a_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_1 & 0 & -a_1 \\ b_2X_2 & 0 & -a_2X_2 & 0 & 0 & 0 & b_2 & -a_2 & 0 \\ c_2X_2 & 0 & 0 & 0 & -a_2X_2 & 0 & c_2 & 0 & -a_2 \\ b_3X_3 & b_3Y_3 & -a_3X_3 & -a_3Y_3 & 0 & 0 & b_3 & -a_3 & 0 \\ c_3X_3 & c_3Y_3 & 0 & 0 & -a_3X_3 & -a_3Y_3 & c_3 & 0 & -a_3 \end{pmatrix}}_{\mathcal{A}} \begin{pmatrix} R_{11} \\ R_{12} \\ R_{21} \\ R_{22} \\ R_{31} \\ R_{32} \\ T_1 \\ T_2 \\ T_3 \end{pmatrix} = \begin{pmatrix} -D_1 \\ -D_2 \\ -D_3 \\ -D_4 \\ -D_5 \\ -D_6 \end{pmatrix} \tag{35}$$

Since the rank of  $\mathcal{A}$  is 6, we compute the variables using three unknowns  $l_1, l_2$  and  $l_3$ :

$$\begin{pmatrix} R_{11} \\ R_{12} \\ R_{21} \\ R_{22} \\ R_{31} \\ R_{32} \\ T_1 \\ T_2 \\ T_3 \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \end{pmatrix} + l_1 \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \end{pmatrix} + l_2 \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{pmatrix} + l_3 \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \\ d_9 \end{pmatrix} \tag{36}$$

To solve for the unknown parameters, we use 3 orthonormality constraints for the rotation variables:

$$R_{11}^2 + R_{21}^2 + R_{31}^2 = 1 \tag{37}$$

$$R_{12}^2 + R_{22}^2 + R_{32}^2 = 1 \tag{38}$$

$$R_{11}R_{12} + R_{21}R_{22} + R_{31}R_{32} = 0 \tag{39}$$

We obtain a final set of equations by substituting all the rotation variables using only three unknowns  $(l_1, l_2, l_3)$  using (36). To efficiently solve the final set of equations derived from the above 3 second-order equations, we used a software for Groebner basis method (Kukelova et al. 2008). A total of 8 solutions is obtained for this problem, which

agrees with the 8 solutions in the case of generalized pose estimation problem (Nistér 2004).

#### 5.4 Pose Estimation using Both 2D-to-3D and 3D-to-3D Correspondences

Classical pose estimation refers to the computation of the pose of an object using three 2D-to-3D point correspondences. We show that one can compute the pose when we are given mixed correspondences. Let us assume that we are given three 3D points  $P_1$ ,  $P_2$  and  $P_3$  in the first reference frame. As shown in Fig. 5(e), let the corresponding geometric entities in the second reference frame be one 2D point and two 3D points  $Q_2$ ,  $Q_3$  respectively. Let the projection ray corresponding to the 3D point be given by  $\mathcal{L}_1$ . This is equivalent to the following correspondences between the geometric entities.

- $P_1$  lies on  $\mathcal{L}_1$ .
- $P_2$  corresponds to  $Q_2$ .
- $P_3$  corresponds to  $Q_3$ .

We transform the geometric entities in the second reference frames to three planes as shown in Fig. 5(e). According to this transformation we can solve the mixed pose estimation using the point-to-plane algorithm for the configuration  $Points(2, 2, 2, ) \leftrightarrow Planes(3)$  where the following constraints should be satisfied after the registration:

- $\Pi_1$  must contain the points  $P_1$  and  $P_2$ .
- $\Pi_2$  must contain the points  $P_1$  and  $P_3$ .
- $\Pi_3$  must contain the points  $P_2$  and  $P_3$ .

## 6 Experimental Results

First we show the experimental results for the different cases of point-to-plane registration algorithms using simulations and real experiments in Sects. 6.1 and 6.2 respectively. For validating the point-to-plane registration algorithm, we conducted two real experiments using a contact sensor and 3D city models. In Sect. 6.3 we validate our theory on generalized registration algorithms using a Kinect sensor.

### 6.1 Simulations

We analyzed the performance of our minimal solutions in simulations by generating 32 random planes inside a cube of side length 100 units. We randomly sampled 320 points on these planes within the cube. A test set was created by transforming all 320 points using a ground-truth rotation and translation, then adding Gaussian noise to each point.

We randomly selected  $k$  points from the test set according to the point-to-plane configuration of the algorithm, then computed the rotation and translation using the points and

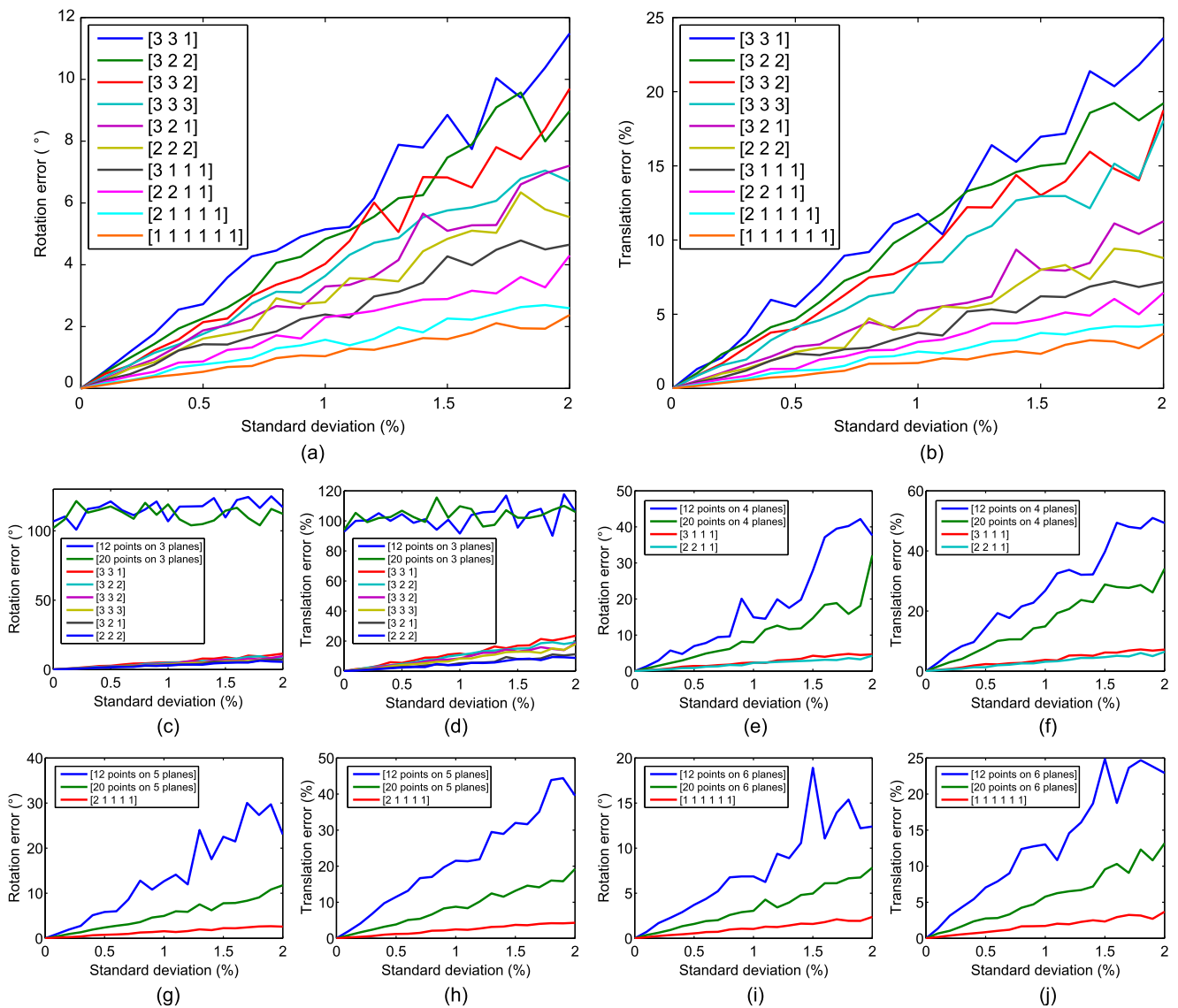
the corresponding planes. The estimated transformation was then evaluated by using it to transform the other  $320 - k$  points and computing the mean point-to-plane distance between the transformed points and their correct corresponding planes. Each trial consists of generating a test set, then repeating the selection of  $k$  points and transformation estimation 100 times for this test set. Of the resulting 100 transformations, the solution for the trial is the one transformation that provides the minimum mean distance.

Figure 8 plots errors in estimated rotation and translation with varying noise levels. For each configuration, the errors plotted are the average of 100 trials. For each number of planes ( $n = 3, 4, 5, 6$ ), we compare our minimal solutions for every possible configuration of 6 points (as well as the non-minimal configurations for 3 planes that were included in Table 1) to a least-squares solution for the same number of planes using 12 or 20 points without orthonormality constraints. In all cases, our minimal solutions yield smaller errors than the least squares method. Note that the least squares method completely fails in the case of three planes. Thus, our transformation is useful not only for the minimal configurations but also in non-minimal configurations such as (3, 3, 3).

### 6.2 Real Experiments

#### 6.2.1 Registration using a Contact Sensor

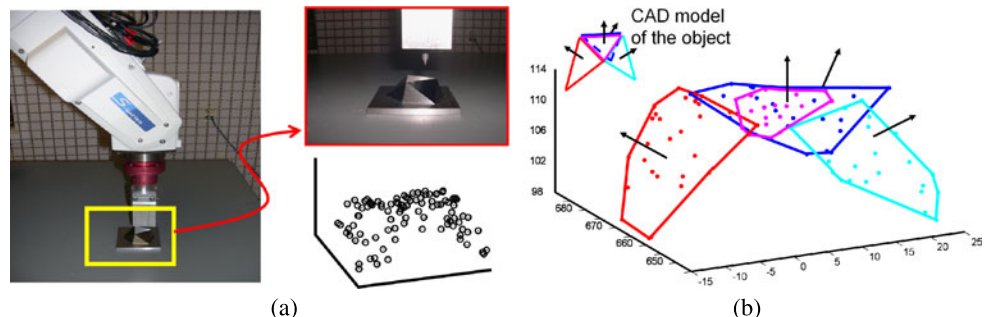
The first experiment, shown in Fig. 9, was conducted using a 6-degree-of-freedom robotic arm with a built-in contact detection function. We used as the target object a partial surface of an icosahedron, of which four of the 20 faces are measurable, as shown in Fig. 9. The robot automatically measured 100 points (contact positions) on the surface; each point was measured by first moving the probe to a random  $x$ ,  $y$  position and then moving down towards the surface (in the negative  $z$  direction) until it sensed a contact. We clustered the points using a simple RANSAC-based plane fitting algorithm. There were four main clusters corresponding to the four planes of the icosahedron used in the experiment. Next, the method described in Sect. 4 was used to find the correspondences between these clusters and the planes in the 3D model. Given these correspondences, we applied our point-to-plane algorithm using several of the minimal 3-plane and 4-plane configurations. As in the simulations, we repeated the following process to determine the solution: randomly selecting  $k$  points, solving for the transformation, and evaluating the mean distance of the transformed remaining points to the 3D model. The final point-to-plane distance error for all of the inliers was about 3 % of the overall size of the scene. The least squares method failed completely for the 3-plane case (similar to the results shown in Fig. 8). In the 4-plane case, the least-squares error was about 10 times larger than the error of the minimal solutions.



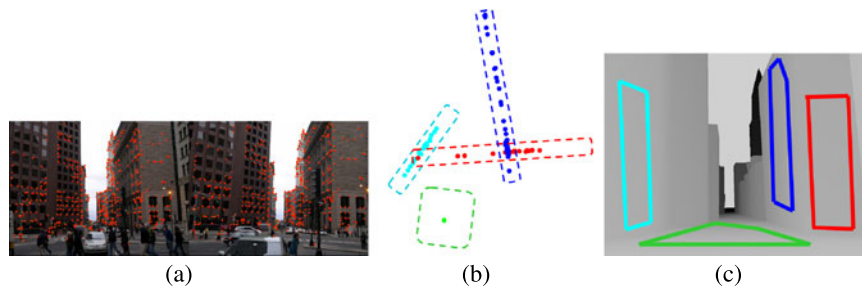
**Fig. 8** Rotation and translation error for simulation data as a function of the level of noise in the test set. The noise standard deviation is expressed as a percentage of the size of the object. The legends list the configurations in order of decreasing error. (a, b) Results from our algorithm for all non-degenerate configurations shown in Table 1. Note that minimal solutions using 6 points provide lower errors than non-

minimal solutions, and solutions for configurations with larger number of planes have lower errors. (b–j) Our minimal solutions compared to least square methods (using 12 and 20 points) for the same number of planes  $n$ : (c, d)  $n = 3$ , (e, f)  $n = 4$ , (g, h)  $n = 5$ , and (i, j)  $n = 6$ . Note that in the 3-plane case (b), least square methods completely fail due to rank degeneracy

**Fig. 9** Real-world experiment with a 6-degrees-of-freedom robotic arm. (a) 3D contact position data were collected for 100 points on the surface using a built-in contact detection function and built-in encoders of the robotic arm. (b) Plane fitting of the 3D points and the correspondences of the points to the planes in the CAD model using the method of Sect. 4

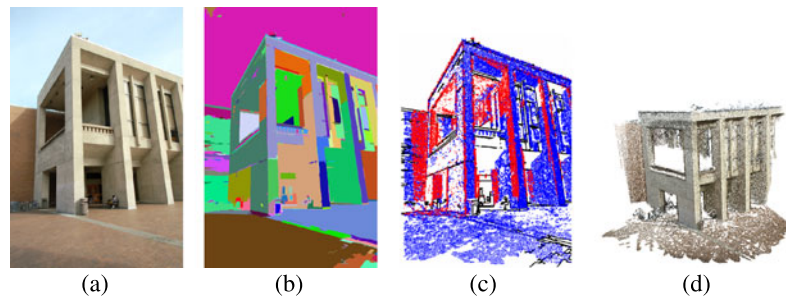






**Fig. 10** (a) An input stereo pair of photos taken in Boston's financial district, overlaid with the points that we matched and reconstructed in 3D. (b) We identify four clusters in the reconstructed 3D points (a sin-

gle point and three planar clouds of points) using a plane-fitting algorithm. (c) The four planes in the 3D city model corresponding to the identified clusters shown in (b)



**Fig. 11** Registering two point clouds, each generated by applying multi-view reconstruction techniques to 15 images. (a) One of the images used in 3D reconstruction. (b) superpixel segmentation of the image shown in (a). (c) The 3D points from the first (*blue*) and second (*red*) clouds are reprojected onto the superpixel image. The points from the first point cloud are used to compute the superpixel plane

parameters, while the second point cloud is preserved as points. The correspondence between the points from the second cloud and the planes obtained from the first cloud are determined by the underlying superpixel. (d) 3D model after merging the two partial reconstructions from the two clusters (Color figure online)

### 6.2.2 Registration of 3D Point Clouds to 3D City Models

Given a plane-approximated coarse 3D model of the city of Boston obtained from a commercial website (<http://www.3dcadbrowser.com/>), we performed localization within the map using a pair of images of a scene in Boston's financial district. To obtain 3D points from the image pair, we matched Harris features and applied standard structure-from-motion algorithms.

Using a RANSAC-based plane fitting algorithm, we fit planes to the reconstructed 3D points. We computed 3 planes from the reconstructed points as shown in Fig. 10. A coarse initialization is manually provided and the nearest planes in the 3D model are identified. All of the planes shown in Fig. 10(c) (more than 10 planes) were used from the 3D model of Boston. Using the method described in Sect. 4, we obtained the correspondences between four clusters (a single point and three planar clouds of points) and four planes in the 3D model. The plane corresponding to the ground had only one 3D point due to occlusion from pedestrians and cars. (Note that it was important to have at least one point on the ground in order to determine the vertical translation.) Applying our minimal algorithms for the 4-planes

case yielded results with an error of just 0.05 % of the overall size of the scene.

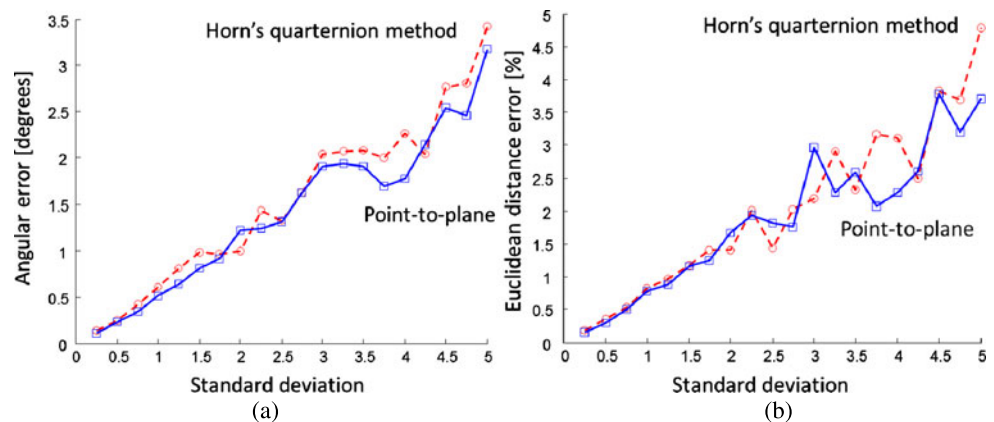
Our point-to-plane registration algorithm can also be used for merging partial reconstructions obtained from multi-view reconstruction techniques (Furukawa and Ponce 2010), as shown in Fig. 11. In order to obtain a 3D model from 30 images provided by Furukawa and Ponce (2010), we subdivide the images into two clusters of 15 images each. We reconstruct 3D point clouds from each image cluster and use the superpixel segmentation of a common image to register them. The 3D points from the first cluster are reprojected onto the superpixel image and used to compute the plane parameters for each superpixel. (We eliminate superpixels with insufficient or non-planar points.) The superpixel segmentation of the common image gives us the correspondences between the points in the second cluster and the planes obtained from the first cluster. We obtain the 3D registration using a RANSAC framework, in which we select three or more non-degenerate planes (see Sect. 3.3) and the corresponding minimum number of points.

Previous work merging partial 3D models obtained from multi-view 3D reconstruction has used non-minimal iterative approaches (Ramalingam and Lodha 2003). However, initializing with a minimal solution, such as the one de-

**Fig. 12** *Top Row:* 6 RGB images are shown from a sequence of 36 frames captured using a Kinect sensor. *Middle Row:* 6 corresponding depth images. *Bottom Row:* Registered 3D model viewed from 3 different viewpoints



**Fig. 13** Comparison between Horn's quaternion approach (red dashed line) and the point-to-plane method (blue solid line) in simulations. (a) and (b) show the rotation and translation error plots respectively (Color figure online)



scribed here, may be critical for noisy 3D data. In addition, there are two general advantages of point-to-plane rather than point-to-point registration: (1) accuracy (Rusinkiewicz and Levoy 2001), (2) compact representation of the 3D models (about a million 3D points are represented using few hundred superpixel planes).

### 6.3 Experiments for Generalized Registration Algorithms

For validating our theory on generalized registration algorithms, we tested the following algorithms in simulations and real experiments:

- 3 point registration using  $Points(3, 2, 1) \leftrightarrow Planes(3)$  configuration
- 3 point registration using  $Points(3, 1, 1, 1) \leftrightarrow Planes(4)$  configuration
- 3 point registration using  $Points(1, 1, 1, 1, 1) \leftrightarrow Planes(6)$  configuration
- Mixed pose estimation using  $Points(2, 2, 2) \leftrightarrow Planes(3)$  configuration

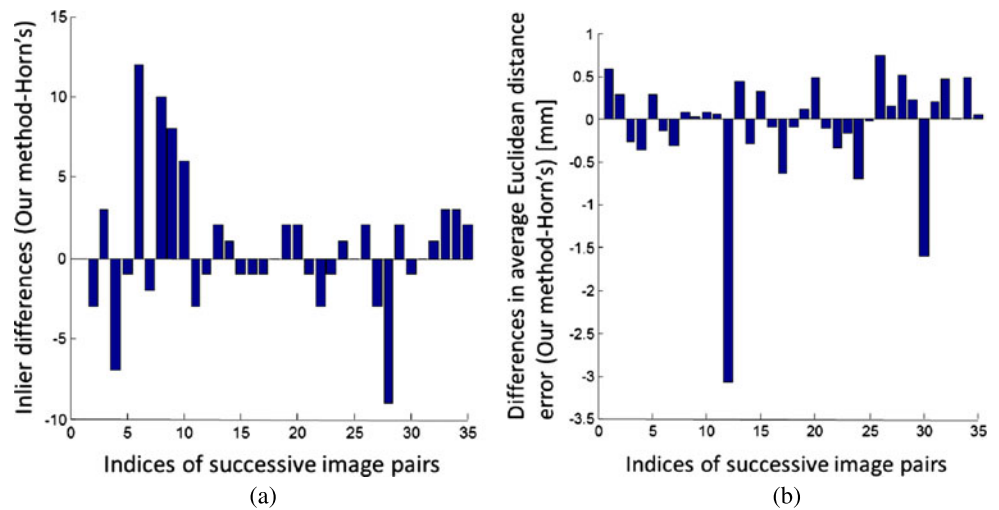
The first three algorithms are based on 3D point correspondences using the three different plane parameterizations

shown in Figs. 5(b), (c) and (d). The fourth algorithm is based on the plane parameterization shown in Fig. 5(e).

In simulations, we compared the performance of the first three algorithms with Horn's quaternion method (Horn 1987). We generated 1000 random points inside a cube of dimension 100 units. The test bed was generated by transforming the points using a ground-truth rotation and translation, then adding Gaussian noise to each point. The different formulations lead to very similar performance with the first case marginally better than the rest. We show the performance of this case with respect to Horn's method in Figs. 13(a) and (b).

We used a Kinect sensor for validating our theory on real data. We captured 36 Kinect image pairs (RGB and depth) for an indoor office sequence as shown in Fig. 12. The calibration information is known for the RGB image and this allows us to compute the ray for every pixel. The ray computation is necessary for testing our mixed pose estimation. We used SIFT features for matching keypoints between images. Using the SIFT correspondences in the RGB image, we obtain 3D point correspondences from the depth image. We observed that by using a hybrid point-to-plane

**Fig. 14** Comparison between Horn's quaternion approach and the point-to-plane method for real data. (a) and (b) show the number of the difference in the number of inliers and average distance error (in millimeters) respectively for the 36 consecutive frames captured using a Kinect sensor



algorithm that chooses different primitives for registration based on our generalized point-to-plane approach, our performance is similar or marginally better than Horn's approach. We compared the average Euclidean distance error between corresponding points and the number of inliers as shown in Figs. 14(a) and (b). During the RANSAC, we used the same triplet for both Horn's approach and the point-to-plane method. Out of 35 successive registrations, the point-to-plane method obtained 22 more inliers compared to the Horn's approach. The average distance error was 2.6 millimeters more for the Horn's method compared to the point-to-plane method. Since the differences are computed for all the 35 registrations, we consider this improvement marginal. We show the 3D model registered from individual Kinect frames in Fig. 12. As in any minimal approach, once the inliers are computed using our point-to-plane algorithm a standard least-squares registration is done between consecutive frames using the selected inliers.

It is interesting to observe that the Horn's quaternion method is not a strictly minimal algorithm. On the other hand, the point-to-plane approach uses only the minimal information from the point correspondences. When we transform the points in the second reference to planes as shown in Fig. 5, we strictly use the minimal information necessary for the registration problem. The point-to-plane based approach can generate a lot more hypotheses for the same set of points. For example, given correspondences between 3 points in two different reference frames, we can only generate one motion hypothesis using Horn's method. However, our method can use different transformation techniques and generate more than 10 different motion hypotheses for the same 3 point correspondences. This is analogous to several minimal problems in vision. For example, the 8-point algorithm can generate a single motion hypothesis for relative motion using 8 correspondences. On the other hand, a 5-point algorithm can generate  $\binom{8}{5}$  different motion hypothe-

ses for the same 8 points. We believe that this ability to generate several meaningful nearby hypotheses is beneficial to find the most optimal solution in a RANSAC framework.

## 7 Discussion

In Table 1, we show that the special handling of individual cases leads to efficient solutions. In addition to this benefit, the specialized algorithms allow us to avoid degeneracy problems. The matrix  $\mathcal{A}$  arising from coplanarity is of size  $6 \times 12$  for the general case  $Points(1, 1, 1, 1, 1, 1) \leftrightarrow Planes(6)$  and this matrix is used in computing the 12 variables in the motion  $(\mathbf{R}, \mathbf{T})$ . The rank of this matrix is 6 and this allows us to use all 6 independent orthonormality constraints to extract the motion parameters. However, the rank decreases to 5 if we use the data from a special configuration  $Points(3, 2, 1) \leftrightarrow Planes(3)$ . So the solution exists in a subspace spanned by 7 unknown parameters. Since we have only 6 independent orthonormality constraints, we will not be able to extract the motion using this approach. On the other hand, the specialized approach constructs a smaller  $\mathcal{A}$  matrix from coplanarity constraints and uses a more compact representation for rotation variables in the equations to avoid the degeneracy problems. This implies that the knowledge of the point-to-plane configuration is necessary to avoid degeneracy problems.

The development of minimal algorithms for registering 3D points to 3D planes provides opportunities for efficient and robust algorithms with wide applicability in computer vision and robotics. Since 3D sensors typically do not perceive the boundaries of objects in the same way as 2D sensors, an algorithm that can work with points on the surfaces, rather than surface boundaries, is essential. In textureless 3D models, for example, it is easier to obtain point-to-plane correspondences than point-to-point and line-to-line correspondences.

**Acknowledgements** Special thanks to Tim K. Marks and Oncel Tuzel in helping us in the conference submission. We would like to thank Jay Thornton, Keisuke Kojima, John Barnwell, and Haruhisa Okuda for their valuable feedback, help and support. We would also like to thank the anonymous reviewers and AC's comments in ECCV 2010.

## References

- Besl, P. J., & McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 239–256.
- Chen, H. H. (1991). Pose determination from line-to-plane correspondences: existence condition and closed-form solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6), 530–541.
- Chen, Y., & Medioni, G. (1991). Object modeling by registration of multiple range images. In *Proc. IEEE int'l conf. robotics automation (ICRA)* (Vol. 3, pp. 2724–2729).
- Dhome, M., Richetin, M., Lapresté, J. T., & Rives, G. (1989). Determination of the attitude of 3D objects from a single perspective view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12), 1265–1278.
- Drost, B., Ulrich, M., Navab, N., & Ilic, S. (2010). Model globally, match locally: efficient and robust 3D object recognition. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)* (pp. 998–1005).
- Enqvist, O., Josephson, K., & Kahl, F. (2009). Optimal correspondences from pairwise constraints. In *Proc. IEEE int'l conf. computer vision (ICCV)* (pp. 1295–1302).
- Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), 381–395.
- Fitzgibbon, A. W. (2003). Robust registration of 2D and 3D point sets. *Image and Vision Computing*, 21(13–14), 1145–1153.
- Furukawa, Y., & Ponce, J. (2010). Accurate, dense, and robust multi-view stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8), 1362–1376.
- Gao, X. S., Hou, X. R., Tang, J., & Cheng, H. F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8), 930–943.
- Geyer, C., & Stewenius, H. (2007). A nine-point algorithm for estimating para-catadioptric fundamental matrices. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)*.
- Grimson, W. E. L., & Lozano-Pérez, T. (1983). Model-based recognition and localization from sparse range or tactile data. *MIT AI Lab, A. I. Memo*, 738.
- Haralick, R. M., Lee, C. N., Ottenberg, K., & Nolle, M. (1994). Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3), 331–356.
- Horn, B. K. P. (1987). Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America, A, Online*, 4(4), 629–642.
- Kukelova, Z., Bujnak, M., & Pajdla, T. (2008). Automatic generator of minimal problem solvers. In *Proc. European conf. computer vision (ECCV)* (pp. 302–315).
- Kukelova, Z., Bujnak, M., & Pajdla, T. (2010). Closed-form solutions to the minimal absolute pose problems with known vertical direction. In *Proc. Asian conf. computer vision (ACCV)* (pp. 216–229).
- Kukelova, Z., & Pajdla, T. (2007). A minimal solution to the autocalibration of radial distortion. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)*.
- Li, H., & Hartley, R. (2005). A non-iterative method for correcting lens distortion from nine-point correspondences. In *Proc. OMNIVIS*.
- Li, H., & Hartley, R. (2007). The 3D–3D registration problem revisited. In *Proc. IEEE int'l conf. computer vision (ICCV)* (pp. 1–8).
- Naroditsky, O., Patterson, A., & Daniilidis, K. (2011). Automatic alignment of a camera with a line scan LIDAR system. In *Proc. IEEE int'l conf. robotics automation (ICRA)* (pp. 3429–3434).
- Naroditsky, O., Zhou, X. S., Gallier, J., Roumeliotis, S. I., & Daniilidis, K. (2012). Two efficient solutions for visual odometry using directional correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4), 818–824.
- Nistér, D. (2003). An efficient solution to the five-point relative pose problem. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)* (pp. 195–202).
- Nistér, D. (2004). A minimal solution to the generalized 3-point pose problem. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)* (Vol. 1, pp. 560–567).
- Nistér, D., & Schaffalitzky, F. (2006). Four points in two or three calibrated views: theory and practice. *International Journal of Computer Vision*, 67(2), 211–231.
- Olsson, C., Kahl, F., & Oskarsson, M. (2006). The registration problem revisited: optimal solutions from points, lines and planes. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)* (Vol. 1, pp. 1206–1213).
- Raguram, R., Frahm, J. M., & Pollefeys, M. (2008). A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In *Proc. European conf. computer vision (ECCV)* (pp. 500–513).
- Ramalingam, S., & Lodha, S. K. (2003). Adaptive enhancement of 3D scenes using hierarchical registration of texture-mapped 3D models. In *Proc. int'l conf. 3-D digital imaging and modeling (3DIM)* (pp. 203–210).
- Ramalingam, S., Taguchi, Y., Marks, T. K., & Tuzel, O. (2010). P2P: a minimal solution for registration of 3D points to 3D planes. In *Proc. European conf. computer vision (ECCV)* (Vol. 5, pp. 436–449).
- Rusinkiewicz, S., & Levoy, M. (2001). Efficient variants of the ICP algorithm. In *Proc. int'l conf. 3-D digital imaging and modeling (3DIM)* (pp. 145–152).
- Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., & Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)* (pp. 1297–1304).
- Snavely, N., Seitz, S. M., & Szeliski, R. (2006). Photo tourism: exploring image collections in 3D. *ACM Transactions on Graphics*, 25(3), 835–846.
- Stewenius, H., Nistér, D., Kahl, F., & Schaffalitzky, F. (2005a). A minimal solution for relative pose with unknown focal length. In *Proc. IEEE conf. computer vision and pattern recognition (CVPR)* (Vol. 2, pp. 789–794).
- Stewenius, H., Nistér, D., Oskarsson, M., & Astrom, K. (2005b). Solutions to minimal generalized relative pose problems. In *Proc. OMNIVIS*.
- Tu, P., Saxena, T., & Hartley, R. (1999). Recognizing objects using color-annotated adjacency graphs. In *Lecture notes in computer science: shape, contour and grouping in computer vision* (pp. 246–263).