

Visual Localisation from Structureless Rigid Models

Guido Manfredi, Michel Devy, Daniel Sidobre

▶ **To cite this version:**

Guido Manfredi, Michel Devy, Daniel Sidobre. Visual Localisation from Structureless Rigid Models. 16th International Conference on Advanced Concepts for Intelligent Vision Systems ACIVS 2015, Oct 2015, Catania, Italy. pp.510-520, 10.1007/978-3-319-25903-1_44 . hal-01355096

HAL Id: hal-01355096
https://hal.science/hal-01355096

Submitted on 22 Aug 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Visual Localisation from Structureless Rigid Models

Guido Manfredi[1,2], Michel Devy1,2, and Daniel Sidobre1,2

[1] CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France,
gmanfred@laas.fr
[2] Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France

Abstract. Visual rigid localisation algorithms can be described by their model/sensor input couple, where model and input can either be 2-D or 3-D sets of points. While Perspective-N-Point (PnP) solvers directly solve the 3-D/2-D case, to the best of our knowledge there is no localisation method to directly solve the 2-D/3-D case. This work proposes to handle the 2-D/3-D case by expressing it as two successive PnP problems which can be dealt with using classical solvers. Results suggest the overall method has comparable or better precision and robustness than state of the art PnP solvers. The approach is demonstrated on an object localisation application.

Keywords: modelling, localisation, motion, structure, PnP, RGB-D

1 Introduction

Visual rigid localisation methods, hereafter shortened to localisation methods, compare a scene's known model to a sensor input and compute the motion between the scene frame and the sensor frame. The model and input are made of 2-D or 3-D points.

Depending on the type of points, 2-D or 3-D, used as model/input, localisation methods can be arranged in four families. In the following, the four families are presented and illustrated through representative works. Table 1 sums up the families and associated works.

In the 2-D/2-D case, the model and input points are 2-D projections of unknown 3-D reference points. The classical approaches are based on finding the fundamental matrix [7], in the uncalibrated case, or the essential matrix [10], in the calibrated case. However, these methods lack precision and they may have mathematical flaws which introduce degenerate cases [1].

The 3-D/3-D methods use two different sets of 3-D points, with different frames, as model and input. Such localisation problem can be solved, for example, with the Iterative Closest Point (ICP) approach [9]. In the ICP method, the pose of a set of 3-D points to another is determined iteratively by matching closest points. These approaches are computationally intensive and building a model with 3-D points can prove challenging, even with 3-D sensors.

According to [11], combining 3-D and 2-D data yields greater robustness and precision than those of the two previous families, so the rest of this work focuses on mixed families.

In the 3-D/2-D case, the model is composed of 3-D reference points and the 2-D input points are projections of the reference points on a camera image plane. This localisation family requires solving the Perspective-N-Point (PnP) problem. Various solutions to this problem will be presented in the next section.

The 2-D/3-D case has gained importance with the advent of cheap RGB-D cameras providing 3-D input. In this case, the model is made of 2-D projections of unknown 3-D reference points, and the input is a set of 3-D points in the camera frame. The principal benefit of such approach is that a single image is sufficient to build the model. The main obstacle stems from the fact that no 3-D frame is initially associated to the scene, thus it is not possible to find a motion between the scene frame and camera frame. To the best of our knowledge, there is no localisation algorithm available to directly solve this case.

This paper proposes a localisation algorithm for the 2-D/3-D case, with a calibrated or uncalibrated sensor. The problem is expressed as two successive PnP problems and solved with classical PnP solvers. The proposed solution robustness and precision are compared with 3-D/2-D localisation methods based on state of the art PnP solvers.

Next section presents the state of the art in PnP solvers. Section 3 introduces the proposed localisation method. An experimental setup is described in Section 4 to compare our approach with classical methods based on PnP solvers. Section 5 presents and discusses the results. Finally, Section 6 concludes this article and opens on future work.

Table 1. Works illustrating the use of the four families of localisation techniques and their hypothesis.

Work	Model	Input	Calibrated
[7]	2-D	2-D	Yes
[10]	2-D	2-D	No
[9]	3-D	3-D	No
[8]	3-D	2-D	Yes/No
Our	**2-D**	**3-D**	**Yes**

2 Related Work

When a set of 3-D reference points, expressed in a reference frame, and their 2-D projections on a camera image plane are available, with known calibration, a PnP solver allows retrieving the motion between the reference frame and the camera frames. Solvers can be separated into iterative and non-iterative methods. This work focuses on non-iterative methods as they are faster and thus more appropriate for real-time applications [8].

Nevertheless, this comparison considers one popular iterative method developed by Lu et al. [6] which computes iteratively the rotation and translation using SVD.

In their work [4], Li et al. introduce the RPnP solver. They propose to divide the 3-D reference points into 3-points subsets, express the problem for each subset as a polynomial and then create a cost function from the sum of squares of these polynomials. The solution correspond to the optimum of this cost function.

Zheng et al. suggest a more precise method dubbed OPnP [14]. In this method, the rotation is expressed as a non-unit quaternion, thus relaxing the optimization problem constraints. The whole problem can then be solved with a Grobner basis solver. The main benefit is that it is a global optimization which can handle any singular case and will find all possible solutions. The main drawback being that when various solutions are available, it is not possible to tell which is the correct one. It is interesting to note that in their results, when various solutions are available, the authors select the solution closest to the ground truth in a L2 norm sense. This is not possible for an actual problem.

Finally, the authors of [3] show that a Direct Least Square (DLS) approach can be applied. They propose to use a Cayley-Gibbs-Rodriguez parametrization for the rotation. Relaxing scale constraints, it is possible to express the scale and translation as a function of the rotation. It is then possible to find the rotation with a least square approach and, from it, compute the translation and scale.

The aforementioned methods work well in the 3-D/2-D case. However, they are not applicable directly in the 2-D/3-D case. The next section shows how a 2-D/3-D problem can be formulated as two successive PnP problems and solved with any of the previous solvers.

3 Localisation with a Structureless Model

In the 2-D/3-D case, no scene frame, nor 3-D points, are initially available in the model, hence the term structureless model. The first step is to create a scene frame S, once and for all, with a first 3-D input. Then, localisation is possible for subsequent inputs.

Consider a set of 2-D points in the image plane of a perspective camera C_1. Then, a depth camera C_2 provides a set of corresponding 3-D points, for example by matching natural features from C_1 and C_2, with coordinates expressed in C_2. The goal here is to find the SC_2 motion. However, as explained earlier, there is no frame S associated to the scene, so localisation is not possible.

In order to make localisation possible, the 3-D points from C_2 are used to arbitrarily define S. By construction, the SC_2 motion is known. Then, the 3-D points from C_2 are expressed in frame S. Thereupon, the scene has an associated frame and 3-D points expressed in it. With corresponding 2-D points from C_1, it is now possible to use a PnP solver to determine the SC_1 motion once and for all.

When a new set of 3-D points is available from a new depth camera C_3, one finds the corresponding 2-D points in C_1's image plane. These 3-D/2-D

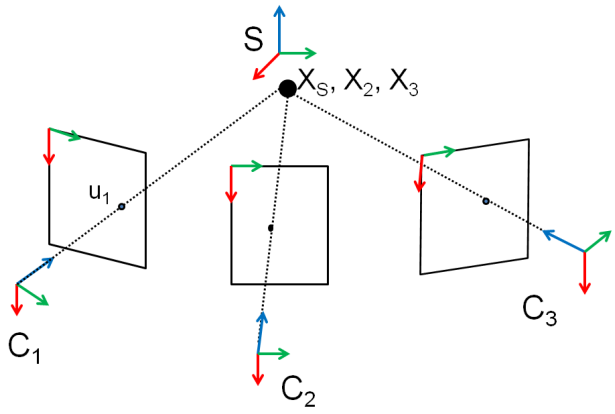


Fig. 1. Illustration of the proposed localisation method. Though various points are necessary for motion computation, for clarity a single 3-D point X_S and its projection u_1 are considered. The point u_1, on C_1's image plane, form the model. The input is made of X_2, the point X_S expressed in C_2, and X_3, the point X_S expressed in C_3. First, a frame S is created from X_2. Then, the motion SC_1 is estimated using u_1 and X_2. Finally, the motion C_3C_1 is computed using u_1 and X_3. With SC_1 and C_3C_1, one can retrieve the desired motion SC_3

correspondences allow solving a PnP problem to obtain the C_3C_1 motion. Since the SC_1 motion is known, computing the SC_3 motion is straightforward. The whole process is illustrated in Figure 1.

From a computational point of view, the process can be split in three steps. First, offline modelling, where 2-D points from the first image plane are computed and added to the model. As only 2-D data is required to build the model, it can be built from any camera or from the Internet, provided calibration data is available. Second, online modelling, where a camera input allows defining a scene frame and computing the SC_1 motion. Third, online localisation, where a new camera input allows computing the motion between the scene and this new camera. In the following, each step is described in more detail.

3.1 Offline modeling

The starting data is a set u_1 of 2-D points. In the pinhole camera model, u_1 verifies the projective relationship,

$$u_1 = KP_1X_S, \tag{1}$$

where K is C_1's intrinsic parameters matrix, P_1 is the SC_1 motion and X_S is the set of 3-D points corresponding to u_1, expressed in the S frame. In this equation, K is known but P_1 and X_S are unknown, they will be determined in the next step.

3.2 Online modeling

A new depth camera C_2 provides a set X_2 of 3-D points expressed in C_2, associated to 2-D points u_1 learned in the previous step. A scene frame is arbitrarily defined at the barycentre of X_2 and for simplicity the SC_2 rotation is set to identity. With P_2 being the SC_2 motion,

$$X_S = (P_2)^{-1} X_2, \tag{2}$$

This equation allows computing X_S, the 3-D points in the scene frame, from X_2. Then any PnP method allows solving Equation 1 to get P_1. The matrix P_1 is saved in the model and represents the SC_1 motion.

3.3 Online localisation

Now that P_1 is known, it is possible to localise a new depth camera C_3, from which a set of points X_3 is acquired. Let's suppose the correspondences between u_1 and X_3 are known, then adapting Equation 2 to this new camera and combining it with Equation 1 gives the localisation formula,

$$u_1 = K P_1 (P_3)^{-1} X_3, \tag{3}$$

where P_3 is the SC_3 motion, the u_1 are known from the offline model, K is known a priori, P_1 is known from the online modelling step and the X_3 are a set of input points expressed in C_3 frame. Again, any PnP method allows computing $P_1(P_3)^{-1}$. Then, P_3 can be directly obtained.

Note that only K, the intrinsic parameters of C_1, are needed. Calibration data from cameras C_2 and C_3 are not necessary.

3.4 Uncalibrated Case

When K is not available, for example when the image is taken from the internet with no EXIF data available, the intrinsic parameters must be determined on the fly. This is done at the online modelling step. Instead of using a PnP solver to recover P_1, an auto-calibration method is used to compute $T_1 = KP_1$. Then, T is decomposed into K and P_1 and the online localisation step can be done normally. In order to compute T_1, the most general approach is the Direct Linear Transform (DLT) [2] which uses a least-square approach to approximate the coefficients of T_1. However, under some hypothesis other methods can be used. For example, if the optical center position can be approximated from the images dimensions in pixels, then Tsai's method provides an efficient solver. Finally, there is a direct relationship which allows recovering exactly K and P_1 from T_1 The precision loss due to the approximation of T_1 is evaluated below.

This approach allows using as model a simple image, calibrated or uncalibrated. It enables to use the billions of images available online to build a partial or full model of almost any textured object.

4 Experimental Setup

To assess the robustness and precision of the proposed localisation method, four simulation experiments are set-up. Two of them estimate the robustness against noise and the precision when the number of corresponding points (u_1,X_2) and (u_1,X_3) vary. The third experiment compares the precision of the uncalibrated solution with the precision of the calibrated one.

In order to simulate data as close as possible to real ones, the 3-D reference points should be realistically projected on camera's image planes. To ensure this, the general idea is to define a cuboid in space from which the 3-D points will be picked. Then, cameras are created at random poses such that they see the whole cuboid. Finally, points are randomly picked in the cuboid and projected with noise onto the camera image planes.

The first step is the computation of the minimum distance z, between the camera and the cuboid's center so that the cuboid is engulfed in the camera's field of view. A maximum distance Z, is chosen arbitrarily while keeping it low enough to avoid the planar case due to distance. To ensure random pose, while keeping the camera oriented to the cuboid, a random point A is picked inside the cuboid and a random point B between the spheres of diameter z and Z centred on the cuboid center. To obtain the camera frame, an orthonormal basis is created from $\boldsymbol{BA}$. This process is done for each required camera, three in this case. Finally, n points are randomly created inside the cuboid and projected on each camera plane with a certain amount of zero-mean Gaussian noise. The points and their projections constitute the simulation data.

In these experiments, the cuboid is a cube of side 2 and $Z = 10$. To be consistent with previous work's experiments [14], in the first experiment n is set to 6 while the noise standard deviation σ varies between 0.5 and 5. In the second experiment, $\sigma = 2$ and n varies between 4 and 15. Each experiment is run 1000 times and the results show the mean error over all runs.

In the real world case, the test object is chosen with an arbitrary shape. The object is modelled from a single image taken with a calibrated RGB camera. This image is described with sift features [5]. The SiftGPU library [13] is used in order to speed up the computation. The matching is done by brute force on gpu and only mutual matches are considered. A Xtion Pro live sensor is used to acquire RGB-D images of the object under different points of view. No calibration data are needed for this sensor. Again, sift features are extracted from the images. All points having a NaN depth are filtered out and the remaining ones are matched against the object's model. The first RGB-D view is used for the online modelling step, this step needs to be done only once. All subsequent views can be used for online localisation, they allow computing the object-camera pose. For online modelling and online localisation, transforms are computed using the EPnP algorithm [8], as it is readily available in OpenCV, and refining the result with an iterative Levenberg-Marquardt. Processing one frame on a consumer grade laptop takes a mean time of 200ms.

For all the experiments, the precision of the proposed methods are compared with the state of the art PnP solvers : RPnP, OPnP, DLS and LHM. For the two

calibrated experiments, our approach is based on solving two PnP problems, both are solved using the RPnP solver, as it is the only non-iterative method providing a single solution. Indeed, OPnP and DLS can provide various solutions with no way to discriminate the correct one. For more insight into the working of our method two results are provided: the online modelling error SOM1 and the online localisation error SOM2. For the two uncalibrated experiments, our approach is based on solving an uncalibrated problem, with Tsai's autocalibration method [12] method and a PnP problem using the RPnP solver, as for the calibrated case.

5 Results and Discussion

5.1 Calibrated Case

The results in Figure 3 and 2 show that SOM1, the modelling error, follows the behaviour of RPnP, the underlying PnP solver. Globally, the error decreases with the number of points available and increases with the noise. Regarding SOM2, the online localisation error, its translation error closely follows RPnP performances. However, when increasing the number of points or the noise, the rotation error is lower than the ones of the other solvers.

One could expect our method to perform at best as well as the underlying PnP solver, which is the case for the translation error. However, the rotation error is significantly lower than the one of the underlying PnP method and lower than the ones of state of the art solvers. This could be explained empirically by the fact that our method uses more data than a single PnP solver: a scene-camera transform, a set of 3-D points and a set of 2-D points, are required for the calculus. Moreover, it seems reasonable to think that when a PnP solver is applied to the result of a previous PnP solver, the result is further refined. Finally, the fact that the frame associated to the scene is the 3-D points barycentre may provide some normalisation to the points coordinates, thus reducing numerical approximations.

5.2 Uncalibrated Case

In the first uncalibrated experiment (Figure 4), the rotation and translation errors remain low, lower than the PnP solver's for $\sigma < 3.5$. Then, for $\sigma > 3.5$, it quickly grows and at $\sigma = 3.75$ it is higher than the error of any other method. It keeps raising as noise increases. In the second uncalibrated experiment (Figure 5), the errors are high with less than twenty points. With twenty points and more, the error quickly diminish to an error lower to the state of the art PnP solver's.

Note that none of these methods use a noise reduction approach, like RANSAC for example. However, the autocalibration method computes a new camera intrinsic parameter from noisy points, i.e. which takes into account the noise. The autocalibration process can be understood as finding the image-to-camera pose

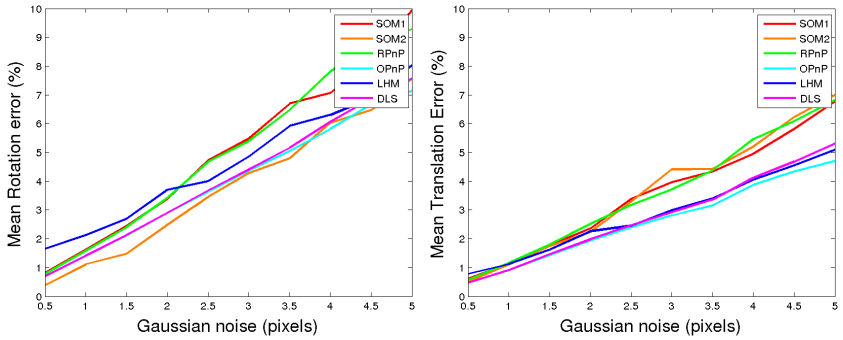


Fig. 2. Mean rotation and translation errors for six points and a noise with standard deviation varying between 0.5 and 5 pixels.

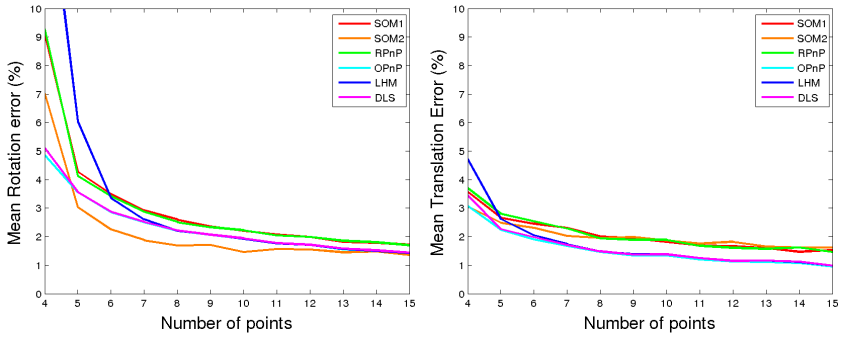


Fig. 3. Mean rotation and translation errors for a noise with standard deviation of two pixels and a number of points going from 4 to 15.

that minimize reprojection error. So the resulting camera suffers from less noise. This can explain why USOM2 has lower error than the other methods. And this points to the fact that autocalibration can be used to diminish noise influence. However, these results also put forward the limits of such methods. Indeed, when there are not enough points ($n < 20$) or too much noise ($\sigma > 3.5$), the autocalibration approach can't find a noise reducing solution.

5.3 Modelling with a Calibrated Camera

To illustrate the proposed method in a concrete object localisation case, we proceed to an experiment on regular objects with a calibrated camera. Figure 6 (a) (b) are the two images needed to build and complete the model. Figure 6 (c) (d) show localisation examples. Note in Figure 6 (b) that the object frame which is set up at this online modelling stage is not necessarily aligned with the object shape. Nevertheless, it allows estimating the object motion as long as the current image matches the offline modelling image.

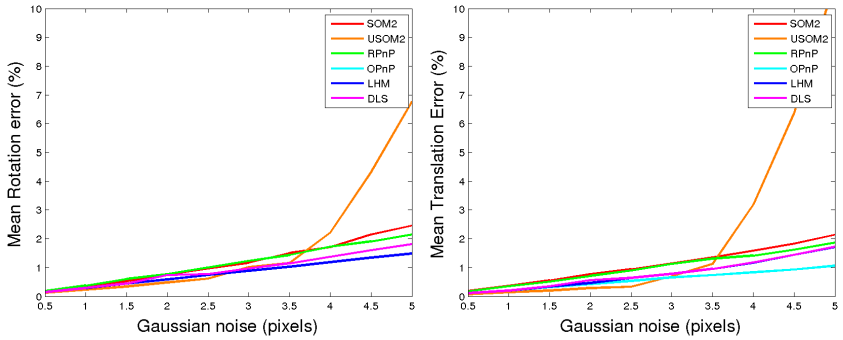


Fig. 4. Mean rotation and translation errors for fifty points and a noise with standard deviation varying between 0.5 and 5 pixels.

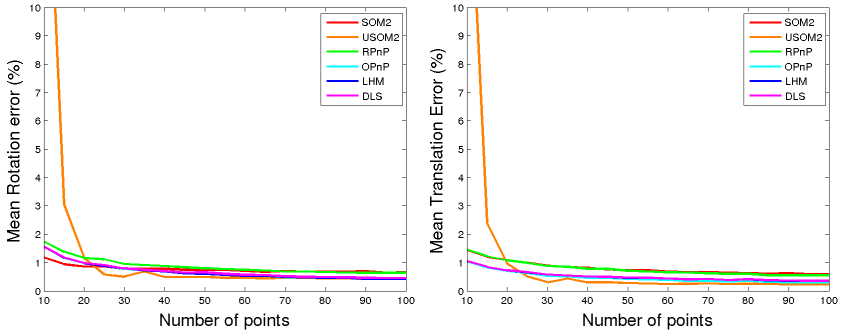


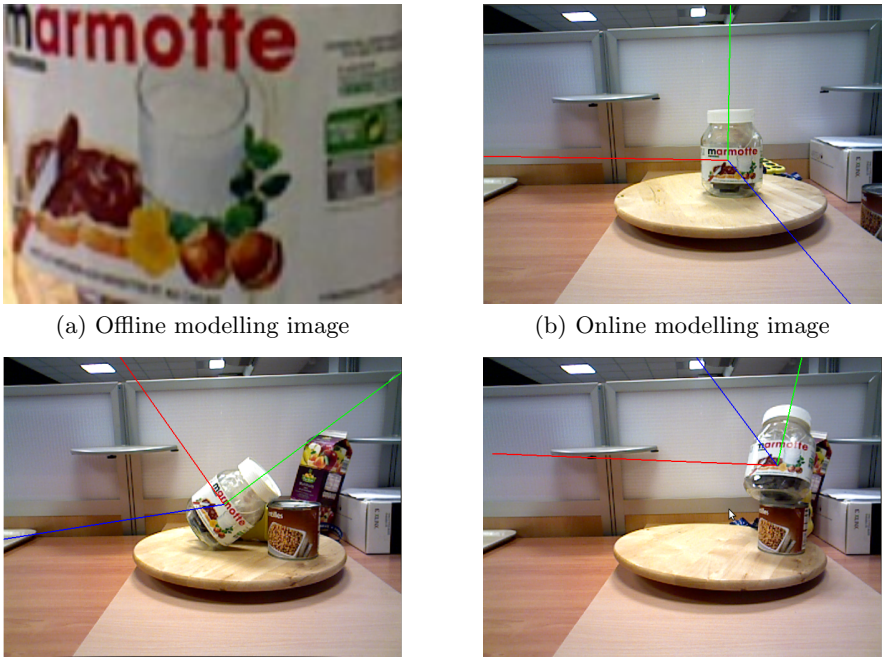
Fig. 5. Mean rotation and translation errors for a noise with standard deviation of two pixels and a number of points going from 10 to 100.

6 Conclusion

This paper tackles the localisation problem when the model is made of 2-D points and the input of 3-D points. Though finding a direct solution seems hard, this work shows that it is possible to express the problem as two PnP problems which can be solved with classical PnP solvers. The benefits of this approach are twofold, the model can be built from calibrated still images only and the overall precision is comparable or better than state of the art PnP solvers. Future work includes bridging the gap between image categorisation and object localisation. Image categorisation learning is done on images, if the same images allow localisation, then it is possible to use advanced categorisation algorithm to recognise objects and then localise them with the presented method.

References

1. Basta, T., Rudas, I., Mastorakis, N.: Mathematical flaws in the essential matrix theory. In: WSEAS International Conference. Proceedings. Recent Advances in



(a) Offline modelling image (b) Online modelling image

(c) Online localisation (d) Online localisation

Fig. 6. Modelling and test images for the marmottela object.

Computer Engineering. WSEAS (2009)
2. Faugeras, O.D., Luong, Q.T., Maybank, S.J.: Camera self-calibration: Theory and experiments. In: Computer VisionECCV'92. pp. 321–334. Springer (1992)
3. Hesch, J.A., Roumeliotis, S.I.: A direct least-squares (dls) method for pnp. In: Computer Vision (ICCV), 2011 IEEE International Conference on. pp. 383–390. IEEE (2011)
4. Li, S., Xu, C., Xie, M.: A robust o (n) solution to the perspective-n-point problem. Pattern Analysis and Machine Intelligence, IEEE Transactions on 34(7), 1444–1450 (2012)
5. Lowe, D.G.: Object recognition from local scale-invariant features. In: Computer vision, 1999. The proceedings of the seventh IEEE international conference on. vol. 2, pp. 1150–1157. Ieee (1999)
6. Lu, C.P., Hager, G.D., Mjolsness, E.: Fast and globally convergent pose estimation from video images. Pattern Analysis and Machine Intelligence, IEEE Transactions on 22(6), 610–622 (2000)
7. Luong, Q.T., Faugeras, O.D.: The fundamental matrix: Theory, algorithms, and stability analysis. International Journal of Computer Vision 17(1), 43–75 (1996)
8. Moreno-Noguer, F., Lepetit, V., Fua, P.: Accurate non-iterative o(n) solution to the pnp problem. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. pp. 1–8 (Oct 2007)
9. Newcombe, R.A., Davison, A.J., Izadi, S., Kohli, P., Hilliges, O., Shotton, J., Molyneaux, D., Hodges, S., Kim, D., Fitzgibbon, A.: Kinectfusion: Real-time dense

surface mapping and tracking. In: Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on. pp. 127–136. IEEE (2011)
10. Nistér, D.: An efficient solution to the five-point relative pose problem. Pattern Analysis and Machine Intelligence, IEEE Transactions on 26(6), 756–770 (2004)
11. Scaramuzza, D., Fraundorfer, F.: Visual odometry [tutorial]. Robotics & Automation Magazine, IEEE 18(4), 80–92 (2011)
12. Tsai, R.Y.: A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. Robotics and Automation, IEEE Journal of 3(4), 323–344 (1987)
13. Wu, C.: SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu` (2007)
14. Zheng, Y., Kuang, Y., Sugimoto, S., Astrom, K., Okutomi, M.: Revisiting the pnp problem: A fast, general and optimal solution. In: Computer Vision (ICCV), 2013 IEEE International Conference on. pp. 2344–2351. IEEE (2013)