Worcester Polytechnic Institute
DigitalCommons@WPI

Computer Science Faculty Publications | Department of Computer Science

6-1-2007

A Quick 3D-to-2D Points Matching based on the Perspective Projection

Songxiang Gu
Worcester Polytechnic Institute

Cliff Lindsay
Worcester Polytechnic Institute

Michael A. Gennert
Worcester Polytechnic Institute, michaelg@wpi.edu

Michael A. King
University of Massachusetts Medical School Worcester

Follow this and additional works at: http://digitalcommons.wpi.edu/computerscience-pubs

 Part of the Computer Sciences Commons

Suggested Citation

Gu, Songxiang , Lindsay, Cliff , Gennert, Michael A. , King, Michael A. (2007). A Quick 3D-to-2D Points Matching based on the Perspective Projection. .
Retrieved from: http://digitalcommons.wpi.edu/computerscience-pubs/46

This Other is brought to you for free and open access by the Department of Computer Science at DigitalCommons@WPI. It has been accepted for inclusion in Computer Science Faculty Publications by an authorized administrator of DigitalCommons@WPI.

WPI-CS-TR-08-04

June 2007

A Quick 3D-to-2D Points Matching based on the Perspective Projection

by

Songxiang Gu, Cliff Lindsay, Michael A. Gennert, Michael A. King

Computer Science Technical Report Series

WORCESTER POLYTECHNIC INSTITUTE

Computer Science Department

100 Institute Road, Worcester, Massachusetts 01609-2280

A QUICK 3D-TO-2D POINTS MATCHING BASED ON THE PERSPECTIVE PROJECTION

Songxiang Gu, Cliff Lindsay, Michael A. Gennert

Worcester Polytechnic Institute

Michael A. King

Umass Medical School

ABSTRACT

This paper describes a quick 3D-to-2D point matching algorithm. Our major contribution is to substitute a new $O(2^n)$ algorithm for the traditional $N!$ method by introducing a convex hull based enumerator. Projecting a 3D point set into a 2D plane yields a corresponding 2D point set. In some cases, matching information is lost. Therefore, we wish to recover the 3D-to-2D correspondence in order to compute projection parameters. Traditionally, an exhaustive enumerator permutes all the potential matching sets, which is $N!$ for N points, and a projection parameter computation is used to choose the correct one. We define "correct" as the points match whose computed parameters result in the lowest residual error. After computing the convex hull for both 2D and 3D points set, we show that the 2D convex hull must match a circuit of the 3D convex hull having the same length. Additionally a novel horizon validation method is proposed to further reduce the number of potential matching cases. Finally, our matching algorithm is applied recursively to further reduce the search space.

Key words: *Convex Hull, Residual Error, Horizon, Calibration*

1. INTRODUCTION

3D-to-2D points matching is still an open topic in computer vision. Projecting a 3D point set into a 2D plane yields a corresponding 2D point set. If the points are identical, we will lose the correspondence information through the projection. On the other hand, if our input data is 3D and 2D point sets and we want to estimate the projection parameters based on the 3D and 2D points' coordinates, we have to know the points correspondence information. The traditional way [6] to acquire the best matching 3D point set is to enumerate all potential matching cases via camera calibration computation. A single matching case yields a set of projection parameters, which we use to project the 3D points into the 2D plane. We then calculate the residual errors, which we define as the difference between the projected 2D points and the input 2D points. We then choose the matching case with minimal residual error as the correct one. To recover a best matching correspondence without any constraint requires a search space of $N!$ for N points.

Even without any extra constraint, we still can shrink the $N!$ searching space with topology information provided by the points set and multi-view projection. In this paper, we introduce a novel points matching algorithm to pick the correct match with $O(2^n)$ complexity as well as solve the perspective projection parameters. In this paper, we use pose estimation [11] to determine a transformation in order to simulate the perspective projections. Therefore, given a set of matching 2D and 3D points, we can compute the projection parameters with the following closed-form calibration method [10] [16]. Once the camera parameters are determined, we can use them to project the original 3D points into 2D and compute the residual error in 2D. If the matching information is correct with minimal distortion, then the residual error would be zero. In the presence of measurement noise, we expect a small, but non-zero residual error. Therefore, an incorrect match will produce a large residual error and can be disregarded.

We show experimental results to validate the correctness of our method. Our experiment builds a relationship between the camera and a Single Photon Emission Computed Tomography (SPECT) system, which is used to collect the radioactivity information. Our 3D points are both retro-reflective and radioactive and we acquire the 2D image of points by camera and the 3D coordinates by the SPECT. Since the 3D information is collected by radioactivity, it is impossible for us to mark the points for the correspondence. Therefore, by inputting the identical 3D and 2D points set into our method, we can show the correct correspondence as well as the camera parameters. We also designed a simulation procedure to study the performance of our method.

The rest of the paper is organized as follows; section two will covers related work, followed by section 3 which is an in-depth explanation of our approach. Then section four outlines our experiments, and section four discusses the conclusion and future work.

1.1. Related Work

The RANSAC algorithm [4] is one of the most popular algorithm for 3D-to-2D points matching. Developed from the traditional ICP adn RANSAC algorithm [14], Fitzgibbon [5] registered the large number of 3D and 2D points by LM-ICP algorithm. Though it works well for a large number of random points, it does not converge to a global optimum neces-

sarily. Furthermore, both algorithms require models in order to compute a match. For our application, we need an optimal correspondence without providing a model and our point number is small. Goshtasby [7] matched point patterns with convex hull edges. However, he did not investigated the complete inherent relationship between the 3D and 2D convex hulls. Cyr [3] introduced a new method to register 3D objects to 2D projections using shape. Kita [8] introduced an iterative searching method for 3D volume data to 2D projection image registration. This method can be adopted for 3D-to-2D points matching. Ryo [9] developed another estimation method that calculates the pose of 3D objects. All of theses methods do not necessary converge to the best match.

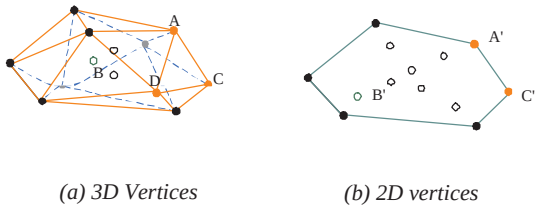
Although calibration is not the main contribution of our work, it is an important factor in this paper. Pin-hole camera model is a close-form calibration model. Tsai [10] introduced a calibration method to solve the pin-hole without distortion and skew. Tsai [16] improves the model by using a 2-step calibration model. With the same input, this method can handle skew and distortion problem. At least 6 points in both 3D and 2D space are required with only a single camera and image. Since we have the 3D and 2D point coordinates and only one camera, we employ Tsai's calibration method in our points matching algorithm.

2. OUR APPROACH

2.1. Overview

In this paper, we combine calibration with points matching computation as a whole to create an efficient algorithm for the global optimized matching case. To simplify the problem, we do not consider distortion in this paper. Although the total potential 3D-2D points matching cases are $N!$, most of the cases can be disregarded. Only the potential matching sets that follow the topology restrictions between 3D and 2D point sets are considered for the calibration computation. To utilize the topology information and simplify the computation, we have to compute the convex hull for both 3D and 2D point sets.

Since we assume no distortion, a 2D convex hull corresponds a circuit on the 3D convex hull(We make a simple proof in Appendix A). With such a theorem, the topology degenerates the factorial method to an exponential one. Secondly, we claim that not all the 3D circuits, but only 3D horizons can be projected into 2D plane as a convex hull. Therefore, we propose a horizon validation method to invalidate a large number of 3D circuits. Since each valid horizon only contains part of the points, the remaining points, which is not on the horizon, could be processed as another point set. Therefore, a recursive method is adopted to search the remaining points dynamically. After putting all the filtered matching cases into the calibration computation, we create a set of camera parameters. Residual error is computed via the



(a) 3D Vertices *(b) 2D vertices*

Fig. 1. (a) Convex hull of 3D point set S; (b) Convex hull of 2D point set S'.

calculated camera parameters. Given a pair of 3D/2D point sets, the exact matching case and the correct projection parameters are finally picked out by the minimum residual error.

The residual error is introduced to measure the validity of a potential matching case. For 3D point set S and 2D projection set S', we have

$$E = 1/n \sum_{i=0}^{n} \|S' - Proj(S, Calib(\bar{S}, S'))\| \qquad (1)$$

Where n is the number of the points; $\bar{S}$ is one of the potential matching cases; $Proj()$ is the projection function with the parameters of 3D point set and camera parameters and $Calib()$ means the calibration. We want to try less times of $\bar{S}$ to get the minimum residual error E.

2.2. Convex Hull Matching

To extract topology information, convex hulls are computed on both 3D and 2D point sets [Fig. 1]. A convex hull of a point set S, is the unique convex polygon or polytope, which contains S and all of whose vertices are points from S [17]. Computing the convex hull is a well studied problem in computational geometry [2]. Yao [18] showed that the lower bound to find convex hulls is $O(n \ln n)$. In two and three dimensions, the quick hull algorithm [1] [15] determines convex hulls for most point sets with time complexity $O(n \ln n)$. However, this method may fail when more than 3 points are coplanar. Finally, O'Rourke [12] provided a $O(n^2)$ robust method for 2D two and 3D convex hull computations.

Based on the convex hull, we have split all the points into two categories: boundary points, which are on the convex hull, and interior points, which are interior the convex hull. For example, in the Fig. 1(a), the 3D point A and C are boundary points and the 3D point B is an interior point. In the Fig. 1(b), the 2D point A' and C' are boundary points and the 2D point B' is an interior point.

Based on two primary theorems *(Theorem 1, Theorem 2 in Appendix A)* from $ComputationalGeometry$ [13] concerning 3D and 2D convex hulls, it is easy to prove that all the

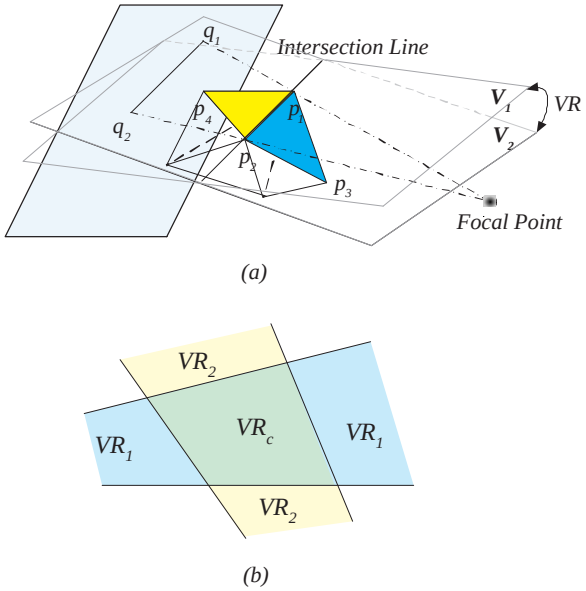


Fig. 2. (a) Valid Region for an convex edge. (b) Common Valid Region intersected by multiple valid regions.

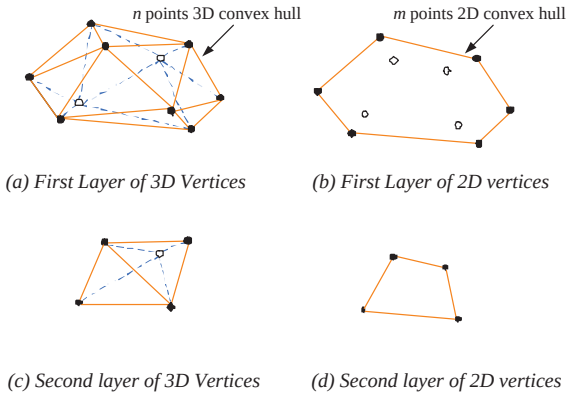


Fig. 3. (a;b) m points circuit in 3D convex hull corresponding to m points in the 2D convex hull. (c;d) Next layer points matching

boundary points on the 2D convex hull correspond to a subset of the boundary points on the 3D convex hull. Furthermore, based on Theorem 3 (Appendix A), we can claim the circuit of 2D boundary points [Fig. 1(b)] must correspond to a circuit of 3D boundary points on 3D convex hull [Fig. 1(a)]. This means, to match the m-length 2D convex hull, we do not have to try all the permutations of the 3D point set, but only the m-length circuits on the 3D convex hull. With the insight from the topology, we shrink the searching space in the first step.

It is not difficult to trace a m length circuit on the 3D convex hull. After tracing such a circuit, if the point number of 2D convex is m, we only need $2m$ trials to search for the correct matching case in the 3D and 2D boundary point sets. In other words, we need m trials to proceed clockwise around the 2D boundary point set and another m trials for the counter-clockwise case. We use the terminology H_n^m to denote the number of valid m length circuits on an n-vertex convex hull.

Considering the constraint on convex hull point matching, we search for a length m circuit on the 3D convex hull for the first matching step, instead of an exhaustive $N!$ search. This first convex hull matching step reduces the computational complexity from $O(N!)$ to $O(2mH_n^m$(N-m)!).

2.3. Horizon Validation

If a 3D circuit could be projected into a 2D plane as a convex hull, all of the points and edges in that circuit should be visible to a certain focus point. Such a circuit is called horizon. In other words, if there is no such a 3D focus point that can view all the edges on the 3D circuit, the circuit is not a valid horizon and therefore can be excluded. Here we developed a horizon validation method to validate the circuits.

As shown in Fig. 2(a), for an edge P_1P_2 in the 3D circuit, there is always a pair of triangles (($\Delta P_1P_2P_3$),($\Delta P_1P_2P_4$)) associated with it. If the edge is projected into a 2D plane as a 2D convex edge, the focus point must view one of the pair of the triangles, but not both. Particularly, in Fig. 2(a), if the focal point is in the region between V_1 and V_2, it can view the triangle ($\Delta P_1P_2P_4$), but not ($\Delta P_1P_2P_3$). Therefore, for each edge, there is a valid region in 3D space that the focus point could only exist in, which we call a "valid region" (VR).

Each edge is determined by its own valid region VR. For two edges, there are two valid regions that can be intersected into a common valid region VR_c (Fig. 2(b)). If there is a focus point that can view these two edges, the focus point has to be in the common valid regions with $VR_c \neq null$. In general, if all of the edges have at least one common valid region ($VR_c = VR_1 \bigcap VR_2 ... \bigcap VR_l \neq null$, l is the edge number), the circuit is proved to be a horizon. Otherwise, if the common valid region is null, no matter where we put the focal point we can not view the circuit as a horizon. Therefore, we consider the circuit to be invalid. Though we have not analyzed the horizon validation mathematically, based on the results of our simulation shown in Section 3.3, this algorithm is an improvement of approximately $O(2^n)$ over an exhaustive search.

2.4. Recursive Computation

As shown in Fig. 3(a;b), during the matching procedure, we dynamically split the point sets into m matched points and $(N-m)$ remaining points in both 3D and 2D point sets by the circuit. Since the two $(N-m)$ remaining point sets can be

considered new an independent potential matching set, we developed a recursive method to deal with the remaining points. If $(N-m)$ is still large, we can compute the convex hulls for 2D, 3D $(N-m)$ point sets and create the potential matching cases for it. As shown in Fig. 3(c;d), we deal with the remaining point sets just like the initial point sets. By recursively repeating the procedure mentioned above until the remaining points number is small enough, we can reduce the potential matching cases to $O(2^r \prod_r m_i \prod_r H_{n_i}^{m_i})$, where r is the number of layers (recursive calls) of point set splitting. It is easy to tell that the algorithm has no computational benefit when the number of the remaining points is fewer than 4. Therefore, we can shrink the problem space until $(N-m) \leq 4$.

Not only the circuit searching but also the horizon validation can be recursively propagated. Initially, the valid region is set to infinite before we begin the horizon validation computation. After the first level horizon validation, we go to the next layer of circuit searching. Since the focal point should not be changed when we search the circuit in next layer, the common valid region of the next layer can only stay inside the common valid region of the previous layer. We can propagate the common valid region of the first level horizon as the initial valid region to the next layer, which is no longer infinite. During the valid region propagation, the common valid region may be split into several pieces by the edges and triangles. Fortunately, most of the pieces turn out to be invalid very quickly.

Since $m \leq n$, the 2D convex hull is the benchmark for each recursive convex matching. To optimize the computation, we can pre-compute all the layers of 2D convex hull. However, when we delete m points from the 3D point sets dynamically, we have to recompute the 3D convex hull for the remaining point sets again and compute a new 3D convex hull for next recursive step.

Finally, for each potential point set, we do a calibration computation of the projection matrix. Then we re-project the 3D points into 2D by perspective projection and calculate the residual error. The points matching case with the smallest residual error is determined to be the correct matching case.

2.5. Missing Points

During the projection, some of the 3D points may be lost caused by overlapped, darkened or out of the image range. Then the 3D point number n is less then the 2D point number m. Currently, our basic idea is that we choose n 2D points from m first to make the point number equivalent between the 3D and 2D point sets. Then we put new point sets with the equal point number into the matching algorithm. Given the time complexity for our algorithm is $O(2^n)$, if there are some missing points in 3D point set, the time complexity is $O(\binom{n}{m}2^n)$.

3. EXPERIMENTS

3.1. Camera Parameters

The pinhole model [10] has 13 parameters including the distortion and skew. For each potential correspondence between the 3D and 2D point sets, we compute the camera parameters that optimally projects the 3D world points into the 2D image. We use a closed-form calibration method that computes the intrinsic and extrinsic parameters by solving the perspective projection equation.

$$\vec{X}^c = Proj(CP, \vec{X}^w) \tag{2}$$

Where $Proj()$ is the projection function, *CP* is the set of camera parameters and $\vec{X}^w$, $\vec{X}^c$ are the point coordinates in 3D world and 2D camera plane, respectively. Camera parameter *CP* can be decomposed into

$$CP = A \cdot P \tag{3}$$

Where $A = \begin{bmatrix} f_x & 0 & IC_x \\ 0 & f_y & IC_y \\ 0 & 0 & 1 \end{bmatrix}$ is the set of intrinsic parameters; f_x and f_y are the scaled focal lengths and IC_x and IC_y are the image center coordinates. The extrinsic parameters are $P = [R|T]$, where R is a 3×3 rotation matrix and T is the translation vector. Given a set of at least 6 pairs of points in both $\vec{X}^w$ and $\vec{X}^c$, we can linearly compute the camera parameters, A and P [6]. After the basic camera parameters are computed, we can re-project the 3D points into 2D plane. Then a second step is performed for the distortion parameters [16]. The computed camera parameters are then used to project each point $\vec{X}^w$ according to Eq 2 and the residual error is computed by Eq 1. We select the camera parameters from the parameter set which generate the lowest residual error.

3.2. Verification Using Real Camera Data

We illustrate this method using a 7 point data set as in Fig. 4. Recalling that at least 6 point pairs are needed to calculate CP, we use 1 extra point to provide redundancy. In this case, the 2D optical data comes from an AXIS PTZ 2130 camera with resolution of 640×480 pixels. As mentioned in Section 3.4, a reasonable estimation for the maximum distortion error is 9 pixels. Then we add radioactivity into the centers of retroflective spheres and put the 7-sphere phantom into SPECT. The 3D data acquired has a resolution of $256 \times 256 \times 256$ voxels. Each voxel is a cube with 2.33mm on each side, for a volume of 12.65mm^3. Since the 3D information is acquired by the radioactivity, all the spheres are identical.

Using brute-force matching, 7 points need $7! = 5040$ calibration computations which completes in 1.26 seconds. In our experiment, we repeat the data acquisition 6 times with

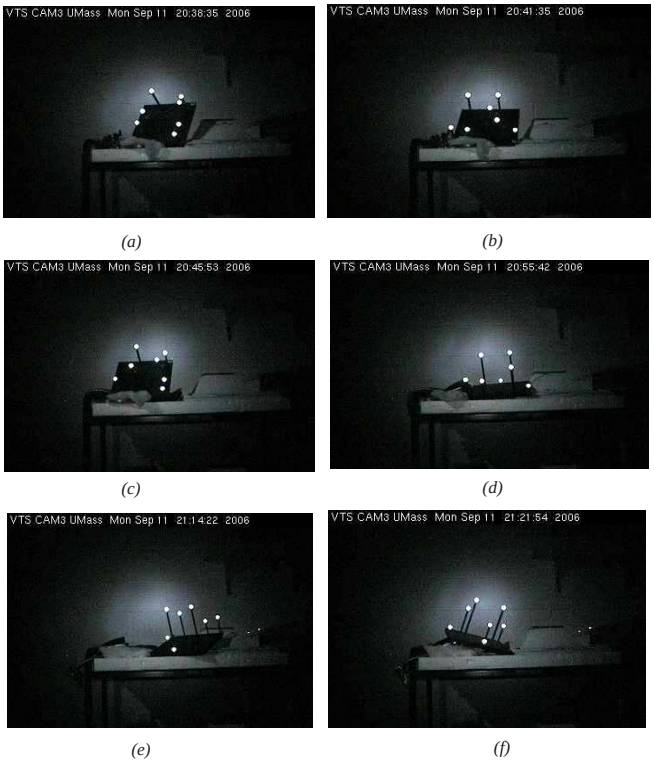


Fig. 4. 6 samples with the same image resolution 640×480 but different camera parameters.

different camera parameters [Fig. 4]. The results for such 6 trials are shown in Table 1.

In the experiments, all final matches are optimal. The average value for the residual error is 2.051 pixels. If we have known the 3D coordinates of the retro-reflective markers, the closed-form method developed by Tsai [16] works well. Moreover, this result shows that the current lens distortion usually has little influence in the imaging procedure and does not change the projection topology. Such results signify that our solution is applicable for the point matching problem even with real cameras. For our solution, the average potential matching cases are 817, which is much smaller than 7!.

In this experiment, we can not change the point number at will. Therefore, we have to design a simulation procedure to show the results with different point number.

3.3. Simulation and Result

Based on the closed-form calibration method, we want to know the average time complexity for our algorithm with points sets that have a different number of points. An evaluation is proposed to simulate experiments with more pseudo-points via the following steps.

1. Generate the coordinates for a set of 3D points.

2. Generate a set of camera parameters.

Trial	*Residual Error (Pixel)*	*Time Consumption (Sec)*	*Potential Matching Cases*
1.	2.48	0.313	780
2.	0.37	0.328	780
3.	0.48	0.343	780
4.	3.24	0.322	780
5.	1.12	0.375	1002
6.	3.18	0.329	780
Mean	1.810	0.335	817.0

Table 1. Time Consumption for the Real Data.

3. Project the 3D points into the 2D camera plane.

4. Shuffle the order of the 2D point set.

5. Put them into our algorithm for the best matching case.

6. Select camera parameters that yield the smallest residual error.

To test our convex hull based points matching algorithm, we generated 8 sets of data with 7 to 14 points, and put them into our algorithm to show the matching results and computation time. Previously, we mentioned a brute-force matching method, a basic convex hull based matching method(Section 3.1) and an improved method with horizon validation(Section 3.2). We compared these three methods to show the average time consumption in the simulation [Fig. 5]. Although we cannot exhaustively list all possible topologies, we repeat the points generation of certain point number with uniform distribution 3000 times. After 8×3000 iterations of the simulation, the mean results are shown in Fig. 5. Fig. 5(a) shows the comparison of the average number of valid potential matching cases and Fig. 5(b) shows the comparison of the average time consumption. From the Fig. 5, our method provides an exponential solution for the point matching problem. Formerly, the brute-force method required 1.26 seconds for 7! calibration computations for optimal matching of 7 points. It is reasonable to state that when the number of points is increased to 13, the brute-force matching takes more than 18 days to calculate making it impractical. Based on the simulation results, only 77.03 seconds on average is required for best matching for 13 points. The time consumption of the method is approximately $O(2^n)$.

Fixing the point number to 9, Fig. 6 shows the time consumption distribution for the 3000 trials with random positions. From that map, it is easy to tell that for most of the random cases, the time consumptions are less than 10 seconds. We also show that the distribution of time consumption is roughly Gaussian. Based on the simulation results, our method decreases the searching time for the best match by the convex hull based enumerator.

4. CONCLUSIONS AND FUTURE WORK

Our algorithm provides a quick method to get the correct 3D-to-2D points matching information. We use the inherent topology information and the multi-view projection principle to shrink the search space. Based on our simulation result, the performance of our method is approximately $O(2^n)$. Our solution fits for some applications, in which the best matching case is required and a small number of points are small. In future, we would like to investigate to use our method with other applications, such as 3D-to-2D points matching, pose estimation and camera calibration. However, it is still an exponential solution. If the point number is further increased, we still have to spend a lot of time on searching the best matching case.

5. REFERENCES

[1] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, 22(4):469–483, 1996.

[2] M. Berg, M. Kreveld, M.Overmars, and O.Schwarzkopf. *Computational Geometry–Algorithm and Applications.second edition*. 2000.

[3] C. M. Cyr, A. F. Kamal, T. B. Sebastian, and B. B. Kimia. 2d-3d registration based on shape matching. In *MMBIA '00: Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, page 198, Washington, DC, USA, 2000. IEEE Computer Society.

[4] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24, June 1981.

[5] A. Fitzgibbon. Robust registration of 2d and 3d point sets. volume II, pages 411–420, Manchester, UK, 2001.

[6] M. A. Gennert, P. P. Bruyant, M. V. Narayanan, and M. A. King. Calibrating optical images and gamma camera images for motion detection. In *Proc. Soc. Nuclear Medicine 47th Ann. Mtg.*, June 2002.

[7] A. Goshtasby and G. C. Stockman. Point pattern matching using convex hull edges. *IEEE Trans. on Systems, Man and Cybernetics*, SCM-15(5):631–637, 1985.

[8] Y. Kita, N. Kita, D. L. Wilson, and J. A. Noble. A quick 3d-2d registration method for a wide-range of applications. *icpr*, 01:1981, 2000.

[9] R. Kurazume, K. Nishino, Z. Zhang, and K. Ikeuchi. Simultaneous 2d images and 3d geometric model registration for texture mapping utilizing reflectance attribute. *ACCV: The 5th Asian Conference on Computer Vision*, 2002.

[10] R. K. Lenz and R. Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3-d machine vision metrology. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(5):713–720, 1988.

[11] S. Linnainmaa, D. Harwood, and L. S. Davis. Pose determination of a three-dimensional object using triangle pairs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(5):634–647, 1988.

[12] J. O'Rourke. *Computational geometry in C*. Cambridge University Press, New York, NY, USA, 1994.

[13] F. P. Preparata and M. I. Shamos. *Computational geometry: an introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1985.

[14] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. *3dim*, 00:145, 2001.

[15] S. Skiena. Convex hull. *8.6.2 in The Algorithm Design Manual*, pages 351–354, 1997.

[16] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *Radiometry*, pages 221–244, 1986.

[17] E. W. Weisstein. Convex hull. *From MathWorld–A Wolfram Web Resource.*

[18] A. C.-C. Yao. A lower bound to finding convex hulls. *J. ACM*, 28(4):780–787, 1981.

Appendix

Theorem 1. [13]

As shown in Fig. 1(b), the line segment l defined by two 2D points A' and C'. $(A'C')$ is an edge of the 2D convex hull($CH(S')$) if and only if all other points of the point set S' lie on l or to one side of it.□

Theorem 2. [13]

As shown in Fig. 1(a), the face p defined by three 3D points A, C and D. ΔACD is a face of the 3D convex hull($CH(S)$) if and only if all other points of the point set S lie on the plane p or to one side of it.□

Lemma 1:

As shown in Fig. 7, given 3D triangle $\Delta v_1v_2v_3$, its 2D perspective projection is also a triangle, called $\Delta v_1'v_2'v_3'$. We also have that all the points interior the triangle $\Delta v_1v_2v_3$ are projected into the interior of triangle $\Delta v_1'v_2'v_3'$.(If the 3D triangle is projected into a 2D plane as a line segment, it can also be considered as a special case of Lemma 1.)

Proof: First of all, we have to prove that a 3D line segment is projected into a 2D line segment with perspective projection. Suppose 3D point coordinates of v_1 and v_2 are (X_1, Y_1, Z_1) and (X_2, Y_2, Z_2); suppose 2D point coordinates of v_1' and v_2' are (x_1, y_1) and (x_2, y_2). From the mathematic expression of the projection, we have

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4}$$

u,v,w are the internal variables; matrix *P* is the projection transformation matrix and the X, Y, Z is the 3D point coordinate. The 2D coordinates can be computed as the following:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} u/w \\ v/w \end{bmatrix} \tag{5}$$

Given a 3D point v_i which is on the line segmentation v_1v_2 with the coordinate $X_iY_iZ_i$, we have

$$\frac{X_1 - X_i}{X_i - X_2} = \frac{Y_1 - Y_i}{Y_i - Y_2} = \frac{Z_1 - Z_i}{Z_i - Z_2} \tag{6}$$

Combined with Eq 4, Eq 5 and Eq 6, we can get

$$\frac{x_1 - x_i}{x_i - x_2} = \frac{y_1 - y_i}{y_i - y_2} \tag{7}$$

where the (x_i', y_i') is the coordinate of 2D point v_i'. If the point v_i' is the 2D projection point of v_i, v_i' is on the line segmentation of $v_1'v_2'$.

Based on the conclusion above, we pick a random point v_i on the line segment v_1v_2. We have known that the projection point v_i' is on the line segment $v_1'v_2'$. Connecting the v_3 and v_i, we can claim that all the points on the line segment v_3v_i are projected into the line segment $v_3'v_i'$. Then we draw a conclusion that all the points interior the triangle $\Delta v_1v_2v_3$ are projected into the interior of the triangle $\Delta v_1'v_2'v_3'$.

A conclusion here is an interior point of 3D convex hull can not be projected to a boundary point of 2D convex hull, $(v \notin CH(S)) \rightarrow (v' \notin CH(S'))$.□

Theorem 3.

Given 2D point set S'[Fig. 8(b)] which is the projection of 3D point set S[Fig. 8(a)], if v_1',v_2' are adjacent points in the 2D convex hull of S', then v_1, v_2 are also adjacent in the 3D convex hull of S, where v_1',v_2' are 2D projections of 3D points v_1 and v_2.

Proof: As shown in Fig. 8, suppose that 2D line $v_1'v_2'$ is a convex edge in S' but the corresponding 3D line v_1v_2 is not a convex edge in S. Choose 3D point v_3 on line v_1v_2 between v_1 and v_2, We have $(v_3 \notin CH(S + \{v_3\}))$. v_3' is the 2D projection of v_3 and lies on the line $v_1'v_2'$. By Theorem 1, we have $(v_3' \in CH(S' + \{v_3'\}))$. However, since we also have $(v' \in CH(S')) \rightarrow (v \in CH(S))$(Lemma 1.), we have a contradiction against the hypothesis $v_1v_2 \notin CH(S)$. Therefore, theorem 3 is proved.□

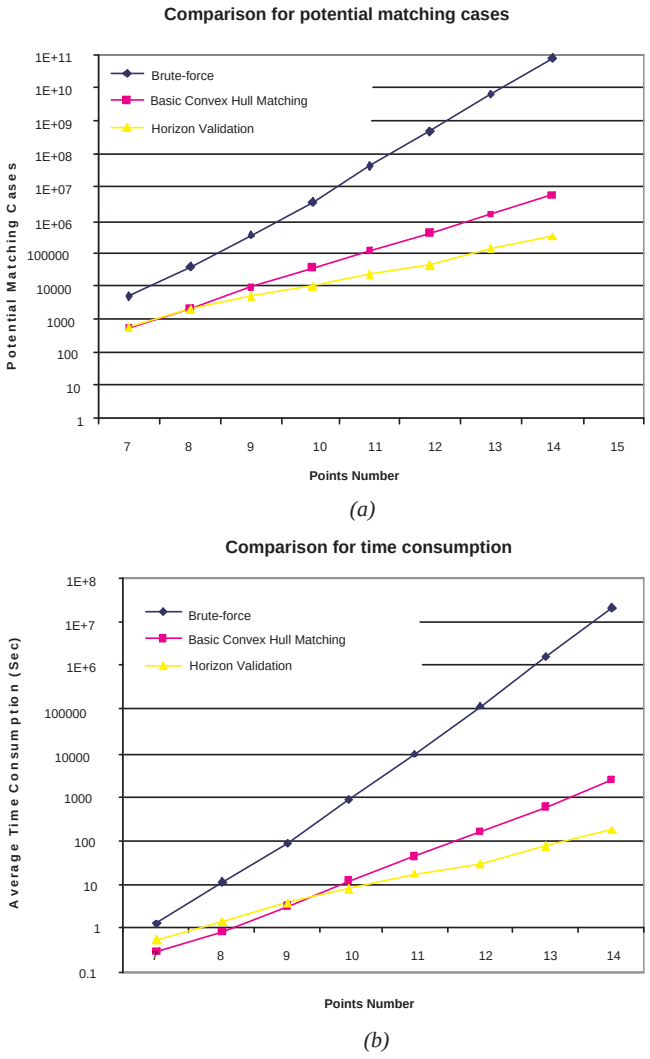


Fig. 5. Comparison between different methods. (a) Comparison of average potential matching cases. (b) Comparison of average time consumption.

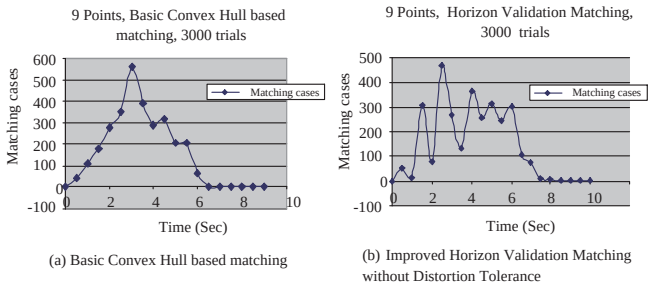


Fig. 6. The time consumption distribution for the 3000 trials of 9 random point sets.

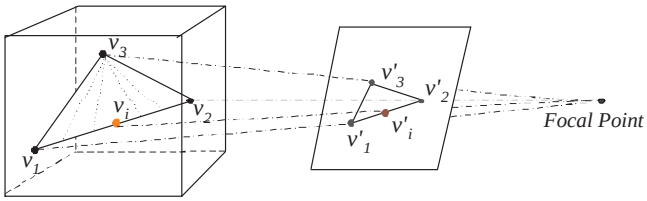


Fig. 7. Triangle Projection.

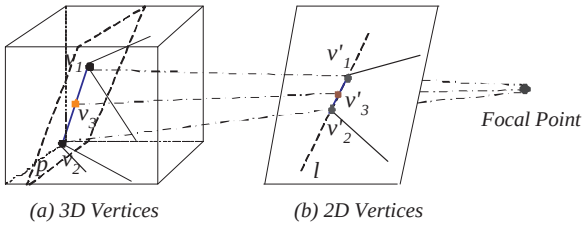


Fig. 8. Corresponding convex hull edge in 3D (a) and 2D (b).