Real-time 3D work-piece tracking with monocular camera based on static and dynamic model libraries

Wenjun Zhu, Peng Wang, Rui Li, Xiangli Nie,

To cite this document:

Permanent link to this document:
https://doi.org/10.1108/AA-02-2017-018

Access to this document was granted through an Emerald subscription provided by esaa

For Authors
If you would like to write for this, or any other Emerald publication, then please use our Emerald for Authors service information about how to choose which publication to write for and submission guidelines are available for all. Please visit www.emeraldinsight.com/authors for more information.

About Emerald www.emeraldinsight.com
Emerald is a global publisher linking research and practice to the benefit of society. The company manages a portfolio of more than 290 journals and over 2,350 books and book series volumes, as well as providing an extensive range of online products and additional customer resources and services.

Emerald is both COUNTER 4 and TRANSFER compliant. The organization is a partner of the Committee on Publication Ethics (COPE) and also works with Portico and the LOCKSS initiative for digital archive preservation.

*Related content and download information correct at time of download.
Real-time 3D work-piece tracking with monocular camera based on static and dynamic model libraries

Wenjun Zhu, Peng Wang, Rui Li and Xiangli Nie
Institute of Automation, Chinese Academy of Sciences, Beijing, China

Abstract

Purpose – This paper aims to propose a novel real-time three-dimensional (3D) model-based work-piece tracking method with monocular camera for high-precision assembly. Tracking of 3D work-pieces with real-time speed is becoming more and more important for some industrial tasks, such as work-pieces grasping and assembly, especially in complex environment.

Design/methodology/approach – A three-step process method was provided, i.e. the offline static global library generation process, the online dynamic local library updating and selection process and the 3D work-piece localization process. In the offline static global library generation process, the computer-aided design models of the work-piece are used to generate a set of discrete two-dimensional (2D) hierarchical views matching libraries. In the online dynamic library updating and selection process, the previous 3D location information of the work-piece is used to predict the following location range, and a discrete matching library with a small number of 2D hierarchical views is selected from dynamic local library for localization. Then, the work-piece is localized with high-precision and real-time speed in the 3D work-piece localization process.

Findings – The method is suitable for the texture-less work-pieces in industrial applications.

Originality/value – The small range of the library enables a real-time matching. Experimental results demonstrate the high accuracy and high efficiency of the proposed method.

Keywords 3D tracking, Automation assembly

1. Introduction

Vision system is widely used in industrial applications such as defects detection (Koch et al., 2015), board manufacturing (Wang et al., 2014), object picking (Richardson et al., 2013) and automatic assembly (Liu et al., 2014; Tang et al., 2014; Schmitt and Cai, 2014; Ahmad et al., 2011). With the development of the three-dimensional (3D) assembly, it is required to improve the intelligence of the automation of assembly as well as the efficiency and flexibility in assembly process. Vision-guided 3D assembly is promising to improve the intelligence, efficiency and flexibility because of the contribution of visual information. It helps to complete the localization, measurement and recognition of the work-piece, which is very important for industrial applications.

The measurement of the 6D pose (i.e. localization) of the work-piece, is essential in the assembly task and can be performed with stereo (Pire et al., 2015; Cvičić and Petrović, 2015), monocular (Forster et al., 2014; Engel et al., 2015) and depth camera (Song and Xiao, 2013). When stereo cameras are used, the 3D structure of the scene can be recovered with geometry constraints. However, 3D reconstruction is time-consuming and an effective depth range is limited by the length of the baseline. When depth cameras are used, there are usually too many noises in the depth image, and the depth range and the size of field of view are also limited. Compared to stereo cameras, the monocular camera is easier to set up in the robotic system which brings more flexibility to the system. Compared to depth images, the color images are in high-resolution and less noisy. Therefore, for this paper, monocular camera is studied for the real-time 3D work-piece tracking problem.

As a state estimation problem, 3D object localization and tracking with monocular camera is a challenging task for its non-linear characteristic. A marker-based (Richardson et al., 2013) tracking system can be utilized in pose tracking, and the points in the marker known as prior can be used to estimate the pose using the perspective-n-point (PnP) methods (Hesch and Roumeliotis, 2011). For example, quick response (QR) code-makers (Olson, 2011) can be used to distinguish the different faces of the work-piece. This method fails if the work-piece is small, as the markers in such cases are difficult to recognize.

Feature-based methods are also widely used for 3D object’s localization and tracking which rely on distinctive features. The correctly matched features provide point...
correspondences which are capable of the measuring the pose with PnP methods. These features are expected to be invariant to viewpoint, illumination changes and also resilient to blur and noise. Additionally, the real time requirement also needs the feature detector and descriptor calculation to be efficient and fast. Point features including SIFT (Lowe, 2004), SURF (Bay et al., 2006), BRISK (Leutenegger et al., 2011) and ORB (Rublee et al., 2011) have good performance for those objects with rich texture, but tend to fail where there are few points that can be detected. Line or edge features (Gomez-Ojeda and Gonzalez-Jimenez, 2016; Zhang and Koch, 2013; Zhou et al., 2015) are good complement when point features are not enough. However, existed line feature descriptors are not distinctive, and line segment detection, similar to the calculation of the feature descriptor, can be time-consuming. To enhance the pose measurement precision, robust method is needed to remove outliers from the false matches (Klein and Murray, 2007; Vacchetti et al., 2004).

Above-mentioned marker-based and feature-based methods are usually faced with problem of occlusion, in which case the point correspondences are not enough to have a robust estimation of the object’s pose. Therefore, view-based method is proposed to tackle this problem. This method matches the tracked image and reference image directly and avoids time-consuming job of the feature descriptor calculation. View-based methods exploit (Younes et al., 2016) all information available in the image and are, therefore, more robust than feature methods for objects with poor texture. Nevertheless, view-based methods are susceptible to failure when scene illumination changes occur as the minimization of the photometric error between two images relies on the underlying assumption of the brightness consistency constraint (Forster et al., 2014; Engel et al., 2015). The photometric model, exposure time and non-linear responding are calibrated in direct sparse odometry (Engel et al., 2016). Another disadvantage is that the calculation of the photometric error at every pixel is computationally intensive. To speed up the matching, semi-direct sparse odometry (Forster et al., 2014) uses the pixel around the features and matches the image patch to obtain the pose. To generate the reference image, Ulrich et al. (2012) used computer-aided design (CAD) model of the object. This method derives a hierarchy of two-dimensional (2D) views of the CAD model. The geometric distance between the image and the derived 2D views is optimized to recognize the work-piece and further recover the pose accurately. This method is robust to noise, occlusions and complex environment but costs much time to complete the recognition.

In this paper, we propose a novel real-time 3D model-based tracking method for work-pieces with monocular camera, which can provide accurate 3D location information of the tracking object continuously. The main contribution of this paper is summarized as follows:

- A dynamic library updating and selection strategy is proposed to speed up the 3D model-based tracking. Pose estimation with dynamic library is implemented by an autoregressive (AR) model. The dynamic library reduces the memory consumption and avoids the whole static global library sustaining in the memory.
- The united view library is quantified into a group of discrete libraries. Each discrete library contains only a small range of the searching space, which significantly reduces the matching time. This result in a coarse estimation of the pose, which is used as an initial value of the optimization to refine the pose measurement.
- The robust of tracking is enhanced by using an adaptive similarity measurement threshold. This strategy is helpful for system’s robustness of occlusion.

The rest of the paper is arranged as follows: Section 2 gives a framework of the proposed method for real-time 3D work-piece tracking with monocular camera. Section 3 details the proposed method and algorithm. Section 4 discusses the performance of the proposed tracking method. Section 5 summarizes the conclusions and prospects.

2. Framework

In this section, the framework of the proposed tracking method will be introduced. Figure 1 shows the overall framework of the proposed tracking method. There are two phases for this method, i.e. the offline phase and the online phase, which are illustrated as follows:

In the offline phase, a static global library, which contains the full range of 2D hierarchy views of the object, is generated based on the intrinsic parameters of the camera and the CAD model of the object. The static global library is composed of a set of discrete hierarchical libraries. Each discrete hierarchical library contains a small range of 2D hierarchy views of the object, and all 2D hierarchical views of the set of the discrete libraries cover full range of poses. The detail of this process will be explained in Section 3.1.

The online phase contains two processes, i.e. the library updating and selection process, and the 3D object localization process. In the library updating and selection process, a dynamic local library is introduced which is composed of specific discrete hierarchical libraries from the static global library. The searching library used for the 3D object localization is selected from the dynamic local library, based on the state (location and pose) of the previous frame. Meanwhile, the dynamic local library is updated. The purpose of updating the dynamic library is to select those discrete 2D views matching libraries, views of which are within a specific range from the pose of the previous frame, and to abandon those exceed the limits. Using the selected matching library, the pose of the current frame can be calculated within a short time in the 3D object localization process, while an adaptive threshold for similarity measurement is generated for each frame. This phase will be explained in Sections 3.2 and 3.3.

3. Algorithm

In this section, the detailed algorithm will be described.

3.1 Global library generation

For view-based localization method, a global model library should be generated firstly. In the matching process, six degrees of freedom of an object in 3D space lead to a huge number of 2D views that must be compared to the image. While an exhaustive search guarantees that the best fitting 2D view is always found, it is much too slow for practical applications. Therefore, most of the view-based methods try to reduce the complexity by pre-computing views of which the camera is placed on the surface of a virtual viewing sphere.
With this approach mentioned above, only three degrees of freedom need to be sampled offline (distance \(d\), latitude \(\varphi\) and longitude \(\lambda\)). The resulting views are compared online to the image, where a remaining degree of freedom must be considered by rotating the 2D view around the optical axis of the camera. Unfortunately, for time-critical applications, there are still too many views that must be transformed and compared to the image. For this reason, hierarchical view-generation method has been studied in Ulrich et al. (2012), and it is used to speed up the localization step in our tracking method (Zhu et al., 2015).

During the generation, a Gaussian sphere is used to generate 2D views. The object’s CAD model is placed at the center of a sphere space, and a virtual camera takes pictures of the work-piece’s CAD model from a certain amount of view points in the sphere. Every view point can be described with three orientations: longitude, latitude and roll angle. Unlike the method in Ulrich et al. (2012), which takes the central pose as zero pose, in our method, the zero pose of the virtual camera coordinates are always fixed to be exactly the same pose as zero pose, in our method, the zero pose of the virtual camera. Unfortunately, for time-critical applications, there are still too many views that must be transformed and compared to the image. For this reason, hierarchical view-generation method has been studied in Ulrich et al. (2012), and it is used to speed up the localization step in our tracking method (Zhu et al., 2015).

During the generation, a Gaussian sphere is used to generate 2D views. The object’s CAD model is placed at the center of a sphere space, and a virtual camera takes pictures of the work-piece’s CAD model from a certain amount of view points in the sphere. Every view point can be described with three orientations: longitude, latitude and roll angle. Unlike the method in Ulrich et al. (2012), which takes the central pose as zero pose, in our method, the zero pose of the virtual camera coordinates are always fixed to be exactly the same pose as zero pose, in our method, the zero pose of the virtual camera. Unfortunately, for time-critical applications, there are still too many views that must be transformed and compared to the image. For this reason, hierarchical view-generation method has been studied in Ulrich et al. (2012), and it is used to speed up the localization step in our tracking method (Zhu et al., 2015).

During the generation, a Gaussian sphere is used to generate 2D views. The object’s CAD model is placed at the center of a sphere space, and a virtual camera takes pictures of the work-piece’s CAD model from a certain amount of view points in the sphere. Every view point can be described with three orientations: longitude, latitude and roll angle. Unlike the method in Ulrich et al. (2012), which takes the central pose as zero pose, in our method, the zero pose of the virtual camera coordinates are always fixed to be exactly the same pose as zero pose, in our method, the zero pose of the virtual camera. Unfortunately, for time-critical applications, there are still too many views that must be transformed and compared to the image. For this reason, hierarchical view-generation method has been studied in Ulrich et al. (2012), and it is used to speed up the localization step in our tracking method (Zhu et al., 2015).

The starting discrete library is \(L_0 \{ [d_{\text{min}}, d_{\text{max}}], [\varphi_{\text{min}}, \varphi_{\text{max}}], [\lambda_{\text{min}}, \lambda_{\text{max}}] \} \); the next one is \(L_1 \{ [d_{\text{min}}, d_{\text{max}}] + \Delta d, [\varphi_{\text{min}}, \varphi_{\text{max}}] + \Delta \varphi, [\lambda_{\text{min}}, \lambda_{\text{max}}] + \Delta \lambda \} \); until the whole searching ranges are covered; and the last one is \(L_{\text{max}} \{ [d_{\text{max}} - \Delta d, d_{\text{max}}], [\varphi_{\text{max}} - \Delta \varphi, \varphi_{\text{max}}], [\lambda_{\text{max}} - \Delta \lambda, \lambda_{\text{max}}] \} \). This kind of discrete library group has a potential defect: when the previous corresponding pose of the work-piece lies on the edge of the library, the corresponding pose of the work-piece may fall out of the searching range the library contains at this circle, and the matching will fail. Steps of the discrete library are introduced to solve this problem. Take the longitude range, for example, the corresponding libraries generated are illustrated in Figure 3.

The overlapped discrete libraries ensure that the corresponding pose of the work-piece never lies on the edge of the chosen discrete library. The persudocode of the static global library generation is shown as Algorithm 1, where the Rodrigues function convert the Euler angle to rotation matrix. \(\pi\) is the camera projection model function:

\[ \pi = \text{Algorithm 1 Global Library Generation} \]

\textbf{Algorithm 1 Global Library Generation} \]

\begin{itemize}
  \item \textbf{input:} \(D\): CAD Model Data
  \item \textbf{input:} \(M\): Camera Intrinsic Parameter Matrix
  \item \textbf{output:} \(L\): Static Global Library
\end{itemize}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{image}
\caption{Overlapped discrete libraries}
\end{figure}
1: for \( d = d_{\text{min}} < d_{\text{max}} \); \( d+ = \Delta d \) do
2: for \( \varphi = \varphi_{\text{min}} \); \( \varphi < \varphi_{\text{max}} \); \( \varphi+ = \Delta \varphi \) do
3: for \( \lambda = \lambda_{\text{min}} \); \( \lambda < \lambda_{\text{max}} \); \( \lambda+ = \Delta \lambda \) do
4: \( R_t = \text{Rodrigues} (\varphi_t, \lambda) \)
5: \( t = d[\cos \lambda \cos \varphi \cos \varphi \sin \lambda] \)
6: \( P = \{ R_t; t \} \)
7: \( I = \pi(D_t, M_t, P_t) \)
8: \( L(d, \varphi, \lambda) = I \)
9: end for
10: end for
11: end for
12: return \( L \)

3.2 Dynamic library updating and selection

In industrial applications, the work-piece always moves consecutively. Thus, previous poses can be used to predict the pose of the current frame. The sequence of these frame poses can be described by a time series. In this subsection, to adapt to the changing appearance and pose of the 3D work-piece during tracking, the online phase is designed, and AR model in time-series analysis is utilized to estimate the pose of the current frame.

We use \( P = \{ P_1, P_2, P_3, \ldots, P_n \} \) to denote the sequence of the time-varying poses in frames. It is known that the AR model can be applied where the series is stationarity. That is to say, the parameters such as the mean and variance of the stochastic process do not change when shifted in time. However, we found that the sequence \( \{ P_t; t \in T \} \) is not stationarity. Therefore, a K-order difference is made on these data to get a new series \( \{ R_t; t \in T \} \). In this paper, K is set to be 3, which is found enough to ensure the stationarity of \( \{ R_t; t \in T \} \). Thus, we have the following:

\[
R_t = -P_1 + 3P_{t+1} - 3P_{t+2} + P_{t+3}
\]  (1)

Then, the AR model of order \( L \) is used in this process as follows:

\[
R_t = \phi_1 R_{t-1} + \phi_2 R_{t-2} + \ldots + \phi_L R_{t-L} + e_t
\]  (2)

where \( \phi_1, \phi_2, \ldots, \phi_L \) are the parameters of the model and \( e_t \) is the white noise process with zero mean \( \mathbb{E}(e_t) = 0 \), constant variance \( \text{var}(e_t) = \sigma^2_e \) and independence assumption \( \mathbb{E}(e_t, e_s) = 0 \). Multiplying \( R_{t+k} \), \( k = 1, 2, \ldots, L \) on the both sides of the above equation and taking the expectation gives the following:

\[
\eta_k = \mathbb{E}(R_t R_{t-k}) = \sum_{i=1}^{L} \phi_i \mathbb{E}(R_{i-t} R_{i-k}) + \mathbb{E}(e_t R_{t-k})
\]  (3)

where \( \eta_k \) is the auto-covariance function of \( R_t \). Thus, the above \( L \) equations can be represented in matrix form as follows:

\[
\begin{bmatrix}
\eta_1 \\
\eta_2 \\
\eta_3 \\
\vdots \\
\eta_L
\end{bmatrix} =
\begin{bmatrix}
\eta_1 & \eta_0 & \eta_{-1} & \cdots & \phi_1 \\
\eta_2 & \eta_1 & \eta_{-1} & \cdots & \phi_2 \\
\eta_3 & \eta_2 & \eta_{-1} & \cdots & \phi_3 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\eta_L & \eta_{L-1} & \eta_{L-2} & \cdots & \phi_L
\end{bmatrix}
\]  (4)

Notice that \( \eta_{-k} = \eta_k \) and:

\[
\eta_0 = \sum_{k=1}^{L} \phi_k \eta_{-k} + \sigma^2_e.
\]

Thus, \( \phi_1, \phi_2, \ldots, \phi_L \) can be solved from the above-mentioned linear equations. Therefore, according to (2), \( R_t \) can be estimated once \( \{ \phi_k; k = 1, \ldots, L \} \) are known. Then, \( P_{t+k} = P_t - 3P_{t+1} + 3P_{t+2} - P_{t+3} \) with \( P_{t+0} \) and \( P_{t+1}, P_{t+2} \) are known. That is to say, \( P_t \) can be predicted from its previous \( L + 3 \) values. In our experiments, \( L \) should be greater than 3 and it is set to be 5, which can ensure an accurate prediction result.

The estimated pose \( P_{\text{estimate}} \) of current frame is used to select a suitable matching library from the dynamic local library. The distance between the estimated pose of current frame and the center poses of neighboring individual discrete library in the dynamic library are calculated. The individual discrete library in the dynamic library with the shortest distance toward the estimated pose of current frame is selected as the matching library in the following 3D localization step.

To overcome the problem that the static library always occupies huge storage space, a dynamic library is set up, which is stored in RAM (random-access memory) and contains libraries not only the selected match library but also the libraries around, see Figure 4. The process of updating dynamic library is to prepare those discrete 2D views libraries, whose center poses are within a specific distance from the estimated pose of the current frame, and to abandon those exceed the limits at the same time.

Pose transformation is also an important step in the library updating and selection process. Every 2D view used for matching
is associated with a pose $H(d, \varphi, \lambda, \phi)$. $\phi$ is the rotation angle around the axis $Z_C$, and $Z_C$ is also the optical axis of the camera. After the pose of the work-piece is determined, which is represented by three pose elements $(\varphi, \lambda, \phi)$, the pose information will be used for guidance for the robot manipulation. A robot’s pose and position is usually expressed by $(x, y, z, \alpha, \beta, \gamma)$, where $(x, y, z)$ is the position of the end effector and $(\alpha, \beta, \gamma)$ are the roll, pitch, yaw angles. The $(\varphi, \lambda, \phi)$ pose should be transformed into $(\alpha, \beta, \gamma)$, so that the following manipulation can be carried out properly. The two pose representations describe two different processes.

1. $(\varphi, \lambda, \phi)$ representation.

Camera is rotated around $X_W$ for angle $\varphi$, then it is rotated around $Y_W$ for angle $\lambda$, and finally, it is rotated around the new $Z_C$ for angle $\phi$.

The final pose is the virtual camera’s pose in the CAD model coordinate system. The two processes expressed in rotation matrix form are as follows:

$$R_x(\gamma)R_y(\beta)R_z(\alpha) = R_x(-\phi)R_y(-\lambda)R_z(-\varphi)$$  \hspace{1cm} (5)

where $R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$ are rotation matrix around $X$, $Y$, $Z$ axis ($c\theta = \cos(\theta), s\theta = \sin(\theta)$).

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta & s\theta \\ 0 & -s\theta & c\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} c\theta & 0 & -s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} c\theta & s\theta & 0 \\ -s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

When the matching completes, $(\varphi_0, \lambda_0, \varphi)$ are the known parameters, and note:

$$R_x(-\phi)R_y(-\lambda)R_z(-\varphi) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

Left side of equation (5) can be expressed as follows:

$$R_x(\gamma)R_y(\beta)R_z(\alpha) = \begin{bmatrix} c\beta c\gamma + s\beta s\gamma c\alpha & -c\beta s\gamma + s\beta c\gamma s\alpha & s\beta c\gamma - s\gamma c\beta c\alpha \\ s\beta c\gamma + c\beta s\gamma s\alpha & c\beta s\gamma + s\beta c\gamma c\alpha & -s\beta s\gamma c\alpha + c\beta c\gamma \\ -s\gamma s\beta & c\gamma s\beta & c\beta \end{bmatrix}$$

As for the position $(x, y, z)$ of the work-piece, obviously, $z = d$. $x$ and $y$ are computed from the image:

$$x = (c_s - c_o) f_x,$$

$$y = (c_t - c_o) f_y,$$

where $(c_s, c_t)$ is the center of the matched work-piece in the frame; $(c_o, c_o)$ is the optical center of the image; and $f_x$ and $f_y$ are pixel scales in width and height. The pseudocode of the dynamic library updating and selection is shown as Algorithm 2, where pyramid denoted the function of generating the image pyramid of input image, AR denoted the AR model function.

### 3.3 Three-dimesional object localization

As shown in Figure 1, once the static and dynamic model libraries are generated, the last step is 3D object localization. In our real-time tracking method, we adopt a hierarchical searching strategy (Ulrich et al., 2012), to speed up view searching process:

**Algorithm 2 Library Updating and Selection**

**input:** L: Static Global Library

**input:** $P_0(k = t - 1, t - 2, t - 3)$: Pose from past frames

**output:** I: Hierarchy of 2D views

1. $P_t = AR(P_0)$
2. $(\phi, \lambda) = Rodrigues(R)$
3. $d = ||d||$
4. $I_t = L(d, \phi, \lambda)$
5. $I = pyramid(I_t)$
6. return $I$

The hierarchical searching process is shown in Figure 5. The circle around the 2D view means a matching process is conducted to the 2D view. While the circle with black edge and light yellow face indicates a match candidate is found, the green line circle means the match is discarded because of low similarity.
During the matching, similarity measures between the 2D views and the current image pyramid are computed using the criterion in Steger (2002):

$$c = \frac{1}{n} \sum_{i=1}^{n} \frac{(m_i,s_i)}{|m_i| \cdot |s_i|}$$  \hspace{1cm} (6)

where $m_i$ ($i = 1, \ldots, n$) are the gradients of the transformed 2D view points, and $s_i$ ($i = 1, \ldots, n$) are the gradients of pixels in the corresponding pyramid of the search image. This similarity measure is invariant to arbitrary illumination changes and is robust against occlusion and clutter.

This measure also has the property that it returns a value between 0 (no similarity) and 1 (perfect similarity) as a score of a potential match. Furthermore, the score roughly corresponds to the portion of the model that is visible in the image. For example, if the object is 50 per cent occluded, the score (on average) cannot exceed 0.5. This is a highly desirable property because it gives the user the means to select an intuitive threshold for when an object should be considered as recognized. In addition, it is intuitive to determine the portion of the occlusion.

The threshold $T_{core}$ is application dependent and should be set to the minimum expected object visibility (see Section 3). In practice, the lower the $T_{core}$ chosen, the more match candidates are generated and tracked through the pyramid, i.e. the slower the matching will be. Meanwhile, the possibility of finding false matches increases with a decreasing $T_{core}$ value. Oppositely, a large $T_{core}$ value speeds up the search process but allows less invisible edges for the object, i.e. some partially occluded objects are not recognized possibly.

In industrial application, the efficiency and robustness is essential. For example, a robot arm need to track work-pieces which may be partially occluded efficiently in assemble task. Thus, the fixed parameter is not suitable and an adaptive threshold is introduced to the proposed tracking method. The main idea is to set a relatively low value for $T_{core}$ while the object is partially occluded, and a relatively high value while the object can be observed totally. Whether the object is occluded can be judged by the variation curve of the previous matching rate values.

In the proposed method, a maximum value $T_{max}$ and a minimum value $T_{min}$ are set. The adaptive threshold value $T_{core}$ for frame $n+1$ is set as follows:

$$T_{core} = \begin{cases} 
C_n - I, & T_{min} < C_n - I < T_{max} \\
T_{max}, & C_n - I \geq T_{max} \\
T_{min}, & C_n - I \leq T_{min}
\end{cases}$$

where $C_n$ is the matching value for frame $n$ and $I$ is an interval value.

Exhaustive matching begins on the highest pyramid level of the hierarchy which only contains several 2D models. Before the matching begins, the 2D models are firstly rotated to cover 360 degrees with an appropriate step, and then scaled with a suitable step to cover the scales adopted in the 2D model generation process which merged the initial views into the current view. The matching will be conducted at each position of the rotated and scaled 2D models in the image. If similarity measure $C$ of the matching exceeds $T_{core}$, a match candidate is considered to appear and the 2D pose of the match labeled as a match candidate. On the next lower pyramid level, the 2D models that have no parent node are searched in the same way like the highest pyramid level. The 2D models on this level that have a match candidate node are also searched, while all the 2D models under an invalid node are discarded. The searching process is repeated until all match candidates have been tracked down to the lowest pyramid level.

After the searching process, all the matches are found in the search image, and their 2D poses (image position, discrete rotation and discrete scale) are determined at the same time. 3D pose $H(d_i, \phi, \lambda, \psi)$ of each match can be computed with the stored corresponding parameters of the 2D matching pose during the 2D views-generation process and the rotation angle of the 2D model in the matching process. But the accuracy of the 3D pose is limited to the sampling of the views and the rotation steps and scaling steps during the 2D matching. Such 3D pose is insufficient for industrial application. Refinement of the 3D pose is conducted using an iterative non-linear optimization using the Levenberg–Marquardt algorithm. The following four steps are taken to complete the 3D pose refinement:

1. 3D CAD model with the obtained 3D pose $H(d_i, \phi, \lambda, \psi)$ is projected into the search image.
2. Corresponding sub-pixel image edge points are searched for each visible projected CAD model edge points.
3. Minimization of the squared distances of the image edge points to their corresponding projected CAD model edge points over the six pose parameters. A refined 3D CAD model is obtained after the minimization.
4. Iteration of (1)–(3), until the correspondences or the refined pose parameters between two iterations no longer change.

After pose refinement, the recognized object with a precise 3D pose can be determined:

**Algorithm 3 3D Object Recognition**

**input:** $I_{fp}$: Hierarchy of 2D views
**input:** $I_{fc}$: Image from Current Frame
**input:** $T_{core,t-1}$: Adaptive threshold updated by last frame
**output:** $P$: Post of the Object
**output:** $T_{core,t}$: Adaptive threshold updated by current frame

The pseudocode of the 3D object localization is shown as Algorithm 3, where match function outputs the best matched 2D view with parameter of $(d_i, \phi, \lambda)$, LM denoted the LM optimization process with the initial pose, Score denoted the score calculation function and threshold denoted the threshold update function.
4. Experiments and results

This section shows the performance of the proposed 3D model-based tracking method. The proposed monocular vision-guided real-time 3D work-piece tracking strategy was evaluated, including the runtime of the search, the robustness against complex background and the robustness against occlusion.

The proposed algorithm is tested on a computer system with the configuration of Quad-Core CPU and 16G RAM. The resolution of the image used for tracking is 640 \times 480. The frame rate of the camera used is set to be 12 FPS.

4.1 Tracking efficiency

Generally, the runtime of the searching process grows with the complexity of the target object because the more complex the target object is, the more features to be matched during the search phase. The searching time in our strategy is independent of the complexity of the object. Because a pyramid template model library was generated and a hierarchical searching strategy was adopted, and the more complex the target object is the more pyramid levels can be generated, and the higher efficiency the searching can be. In addition, because the searching is conducted in the dynamic library, which is determined by time-series analysis, runtime of the tracking is also independent of the pose range of the target object.

A metal bracket was used for the evaluation of the tracking efficiency of our proposed tracking strategy. Videos of the metal bracket with different pose ranges and in different lighting conditions were tested for the tracking efficiency. The average tracking time for a single frame is about 70 ms, which is sufficient for ordinary industrial tasks.

The full possible searching ranges of the object are specified as follows: the longitude $\lambda$ is from 100 degree to 240 degree; the latitude $\phi$ is from 200 to 360 degree; and the minimum and maximum distance of the camera to the object is from 48 to 58 cm. Range intervals for the discrete hierarchical libraries are correspondingly specified as the longitude interval $\Delta \lambda$ is 10 degree, the latitude interval $\Delta \phi$ is 10 degree, the distance interval $\Delta d$ is 2cm. The step of each parameter is set to be half of the corresponding parameter value. Table I shows the specified parameters and the corresponding intervals.

In this experiment, the object is moved by hand. For each frame, the location and pose of the object can be calculated efficiently. Images of the bracket during the tracking are shown in Figure 6. The pose of the matched bracket is visualized by the blue edges, and the consuming time of each frame is listed on the top left of the image. The result shows that the pose of the bracket can be recognized accurately with the proposed method. The consuming time for each frame is coarsely 50~70ms. The three position trajectories of its CAD model center along $x$, $y$ and $z$ axis are shown in Figure 7 (a)-(c), pose trajectories around $x$, $y$ and $z$ axis of the referenced coordinates are shown in Figure 7(d)-(f).

From Figure 6, we can see the bracket is tracked with good accuracy. Even when it is hard to distinguish the position of the hole on the bracket by eyes, the pose of the bracket was still determined precisely. Because the bracket is tracked continuously in our tracking strategy, and the pose of the bracket is not reversed abruptly. See Figure 7, pose of the bracket, namely, the longitude, the latitude and the rotation angle around the optical axis of the camera are all changing continuously during the tracking process. Comparison of tracking time with our method and Ulrich et al.'s (2012) method is listed in Table II.

4.2 Complex background

Our tracking strategy with static and dynamic model libraries shows good robustness against complex background. As the selected dynamic library for the matching only contains a very small pose range, mismatch of the work-piece is not likely to happen. Only the tracked work-piece can be matched with such a library with the small pose range.

Generally, we will not set the matching score thresh value to be very high in the complex background situation. In our experiments, the matching score thresh value was set to be 0.7. Images of the work-piece tracked in the complex background are shown in Figure 8.

In Figure 9, we can see that many disturbing edges appear in the image, some are likely to be similar with the work-piece...
edges. The work-piece can be tracked correctly without mismatch while using our tracking strategy. But mismatch happened when method in Ulrich et al. (2012) was applied. The reason is that the strategy in Ulrich et al. (2012) searches the pose in large range, other pose with similar view will be recognized by mistake, especially in complex environment.

Figure 9 shows the mismatched cases, in which, holes on the bracket was located on opposite side.

4.3 Adaptive matching score threshold
An adaptive matching score evaluation process is used to cope with the occlusion situation during the tracking. When the work-piece is occluded the matching score value will be smaller since some of the matching edge points are missing. If the matching score thresh value is set to be big, the matching will fail if the work-piece is occluded to some degree. If the matching score thresh value is set to be small, wrong objects can be mismatched. To cope with the occlusion situation, we proposed an adaptive matching

Table II  Runtime of the tracking

<table>
<thead>
<tr>
<th>Method</th>
<th>Maximum (ms)</th>
<th>Minimum (ms)</th>
<th>Average (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ulrich et al.’s (2012)</td>
<td>3403.8</td>
<td>783.5</td>
<td>1130.6</td>
</tr>
<tr>
<td>Ours</td>
<td>75.1</td>
<td>45.7</td>
<td>67.4</td>
</tr>
</tbody>
</table>

Notes: (a) x direction; (b) y direction; (c) z direction; (d) latitude; (e) longitude; (f) camroll
score thresh value strategy, which is triggered by continuous diminishing situation of the matching score value. As the work-piece moves in a continuous way, the occlusion also happens continuously, so the portion of the occluded part grows or diminished in a consecutive way.

The trigger of the adaptive matching score thresh value strategy is as follows:

\[ T_2 > T_1 > T \text{ and } T - T_s < T_s \]

\[ T_2, T_1, T, T_s \]

\( T \) is the matching score of the work-piece in the current frame, \( T_1 \) is the matching score in the last frame and \( T_2 \) is the matching score in the frame before last frame. \( T_s \) is the matching score thresh value, and \( T_s \) is a value set to ensure that the matching score value is not too close to the matching score thresh value.

When the adaptive matching score thresh value strategy is triggered, the matching score thresh value will be set to be as follows:

\[ T_{sa} = T_s - (T_2 - T) \]

\( T_{sa} \) is the new matching score thresh value.

In our experiments, the beginning matching score thresh value was 0.7, and the \( T_s \) was set to be 0.02. During the occlusion process, the matching score value changed from 0.7 to 0.58, and the work-piece was tracked successfully using the adaptive matching score thresh value strategy. Images of the process are shown in Figure 10.

The matching score value in the tracking process is shown in Figure 11.
Figure 11 Images of the bracket during the occlusion process

From Figure 11, we can see that the matching score value diminishes when the work-piece is occluded. If the matching score thresh value is set to be constant, the tracking will fail when the matching score becomes smaller than the matching score thresh value. The red line is the matching score thresh value, and it changed when the occlusion happened. With our adaptive matching score thresh value strategy, the matching of the bracket is insured even when it is occluded.

5. Conclusions and future work

A novel method of real-time 3D tracking of work-piece based on CAD model with monocular camera is proposed. Three processes are designed in the proposed method, i.e. the offline global model library-generation process, the online dynamic library updating and selection process and the 3D work-piece localization process. The method is suitable for the texture-less work-pieces in industrial applications. The experiment results show that only 60–70 ms is needed for each frame of the tracking process of the object. The proposed framework is not suitable for the situation when the object miss for a while, that is a problem to solve for our future work.

References


**Further reading**


**Corresponding author**

Peng Wang can be contacted at: peng_wang@ia.ac.cn