D-PET: A Direct 6 DoF Pose Estimation and Tracking System on Graphics Processing Units

Hung-Yu Tseng, Po-Chen Wu, Yu-Sheng Lin, and Shao-Yi Chien
Media IC and System Lab
Graduate Institute of Electronics Engineering
National Taiwan University
Taipei, Taiwan
{hytseng,pcwu,johnjohnlys}@media.ee.ntu.edu.tw, sychien@ntu.edu.tw

***Abstract*—Real-time recovering an accurate 6 DoF pose of a known planar target is essential for augmented reality and robotics applications. Despite several pose estimation tracking systems have been proposed over recent years, there is still the need for a more efficient and more accurate solution for general planar objects. In this work, we develop an innovative GPU implementation of a real-time pose estimation and tracking system. It consists of a pose estimation unit and a pose tracker unit. While the former computes an initial pose of a target using direct method, the latter realizes accurate pose tracking with a hierarchical search scheme. Experiments on both synthetic and real datasets demonstrate that the proposed algorithm performs favorably with various planar targets. By implementing our method on an embedded GPU, the system achieves to work at 11 FPS and is suitable for real-time applications.**

Keywords*—6 DoF Pose Estimation, 6 DoF Pose Tracking, GPU Acceleration

I. Introduction

Tracking the 6 degrees of freedom (DoF) pose (i.e., the position and orientation) of a planar target is one of the main research areas in computer vision. With the development of augmented reality (AR) and robotics, the demand for obtaining accurate and stable 6 DoF poses for general planar targets becomes increasingly vital. Real-time system performance is also necessary for these emerging applications. In the past few decades, numerous systems such as [1] have been constructed for planar targets with binary pattern, which is called fiducial marker. Although theses systems are very efficient, they are not applicable in general cases due to the limited type of planar targets.

Feature-based methods are the state-the-art solutions. The core idea is to find n correspondences between the target image and the camera image using feature matching and tracking algorithms [2]–[5], followed with a RANSAC-based scheme [6] which enhances the reliability of the correspondences. Finally, the Perspective-n-Point (PnP) [7] algorithm is applied to estimate the final pose. Systems applying feature-based method are proposed in recent years [8], [9]. Nonetheless, these systems fail to give the correct result when the target image is textureless, the camera image is blurry, or the tilt angle between the camera and the target is high because the correct feature correspondences they found are insufficient. Recently, Tseng *et al.* [10] propose a direct method to address these issues. Instead of finding point correspondences, the algorithm estimates the 6 DoF poses by measuring appearance distance between the projected target image and the camera image. The

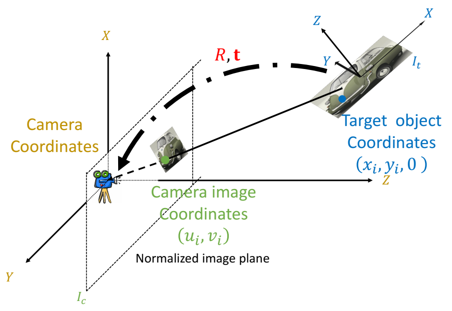


Fig. 1: Coordinate system between target object and camera image.

result shows that it outperforms the feature-based methods in terms of accuracy and robustness. However, the runtime of this method is relatively long due to the large amount of poses needed to be considered. Fortunately, since the computation for each pose is identical and independent, the algorithm has the potential to be executed in highly parallel, which inspires this work.

In this paper, we utilize the parallel characteristic of the graphics processing unit (GPU) to develop an embedded real-time direct 6 DoF pose estimation and tracking system. The main contribution of this work is summarized as follows. First, considering the architecture of GPU, we modify the approximated pose estimation scheme proposed in [10] and propose a 3-scale pose search method to develop the system. Second, the implementation on NVIDIA embedded GPU has accurate performance and achieves real-time computation. Moreover, it is more robust than state-of-the-art feature-based systems.

II. Problem Formulation

The goal of 6 DoF pose estimation is to determine the six degrees of freedom, which is parameterized based on the orientation and position of the target object with respect to the calibrated camera. The problem can be illustrated in Figure 1. Given a planar target image I_t, a camera image I_c, a set of 3D coordinates of points $\mathbf{x}_i = [x_i, y_i, 0]^\top, i = 1 \ldots, n, n \geq 3$ in target object coordinate and a set of image coordinates $\mathbf{u}_i = [u_i, v_i]^\top$ in camera image coordinate. The transformation between these two sets of points can be formulated as

$$\begin{bmatrix} hu_i \\ hv_i \\ h \end{bmatrix} = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & x_0 \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{R}|\mathbf{t}] \begin{bmatrix} x_i \\ y_i \\ 0 \\ 1 \end{bmatrix}, \tag{1}$$

```
Algorithm 1 Approximated Pose Estimation
Input: Target image $I_t$, camera image $I_c$, intrinsic parameters,
    and precision parameters $\varepsilon^*$.
Output: Estimated pose result $\mathbf{p}^*$.
 1: Create an $\varepsilon$-covering pose set $\mathcal{S}$.
 2: Find $\mathbf{p}_b$ from $\mathcal{S}$ with $E_a^{'}$ with random pixel sampling;
 3: while $\varepsilon > \varepsilon^*$ do
 4:     Obtain the set $\mathcal{S}_L$ according to (4).
 5:     Diminish $\varepsilon$;
 6:     Replace $\mathcal{S}$ according to (5).
 7:     Find $\mathbf{p}_b$ from $\mathcal{S}$ with $E_a^{'}$ with random pixel sampling.
 8: end while
 9: $\mathbf{p}^* = \mathbf{p}_b$.
10: Return the pose $\mathbf{p}^*$.
```

where (f_x, f_y) is camera focal length, and (x_0, y_0) is camera principle point. Both are treated as known factors. On the other hand, $\mathbf{R}$ and $\mathbf{t}$ refer to rotation matrix and translation vector, respectively, are the targets of this problem. It is worthy to note that in this problem we aim to pursuit absolute 6 DoF pose, which is different to Visual Odometry [11] that only obtain relative 6 DoF pose between video frames.

III. Related Work

Direct 6 DoF pose estimation [10], which is a two-step algorithm, aims to find the pose $\mathbf{p} = (\mathbf{R}, \mathbf{t})$ that minimizes the following error function

$$E_a(\mathbf{p}) = \frac{1}{n_t} \sum_{i=1}^{n_t} |I_c(\mathbf{u}_i) - I_t(\mathbf{x}_i)|, \tag{2}$$

where n_t represents the total number of pixels in I_t. Approximated pose estimation (APE) is applied in the first stage to obtain the pose. The core concept is to perform a coarse-to-fine estimation in a ε-covering a set of poses. In a ε-covering set, any two poses $\mathbf{p}_1$, $\mathbf{p}_2$ must meet the following constraint

$$\forall \mathbf{x}_i \in I_t : d(T_{\mathbf{p}_1}(\mathbf{x}_i), T_{\mathbf{p}_2}(\mathbf{x}_i)) = O(\varepsilon), \tag{3}$$

where $T_{\mathbf{p}}$ is the transformation at pose $\mathbf{p}$ and ε is the precision parameter in the pose space.

The flow of APE is illustrated in Algorithm 1. A set of poses $\mathcal{S}$ is constructed initially with a coarse precision parameter ε. After obtaining the best pose $\mathbf{p}_b$, the poses within a threshold

$$\mathcal{S}_L = \{\mathbf{p}_L \mid E_a(\mathbf{p}_L) < E_a(\mathbf{p}_b) + L\}, \tag{4}$$

is selected to be processed in the next round, where L is a constant set empirically. Based on the remaining poses, we create a set with finer $\varepsilon^{'}$,

$$\mathcal{S}^{'} = \{\mathbf{p}^{'} \mid \exists \mathbf{p}_L \in \mathcal{S}_L : (3) \text{ holds for } \mathbf{p}^{'}, \mathbf{p}_L \text{ and } \varepsilon^{'}\}. \tag{5}$$

This progress repeats until we obtain the desired precision parameter ε. The algorithm is accelerated by random sampling a portion of pixels instead of using all pixels in I_t to compute (2).

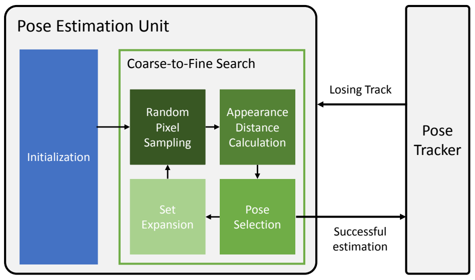


Fig. 2: State chart of proposed D-PET system.

IV. Proposed System

The proposed system is built on NVIDIA Jetson TX1 and consists of two parts, as shown in Figure 2. The pose estimation unit (PEU) is in responsible for finding the initial pose and re-computing the pose when the pose tracker loses the track. The pose tracker (PT) takes the initial estimated pose and performs tracking with the proposed 3-scale search scheme. The details of PEU and PT are described as follows.

A. *Initialization*

The initial ε-covering set described in Algorithm 1 is created at this stage. The origin set construction method is not parallelizable since it determines every parameter sequentially. However, according to [10], we observe that the step sizes of θ_{z_t}, θ_{z_c}, t_x and t_y are all constant given a pair of t_z and r_x, which gives us a chance to utilize parallel computing. The proposed solution is to determine the combinations of t_z and r_x sequentially first, and construct the subset for each combination in parallel afterward. Moreover, the six pose parameters are stored in a single grouped 4 floats and single grouped 2 floats structure, as shown in Figure 3(a). The reason is that the global memory instructions in GPU support reading or writing with 2 floats or 4 floats. This memory allocation makes the system more efficient by reducing the number of global memory instructions.

B. *Coarse-to-Fine Estimation.*

The process is divided into four parts. The random pixel sampling and the appearance distance calculation are respectively in charge of Steps 2 and 7 in Algorithm 1. The pose Selection is responsible for Step 4 while the set expansion accomplishs Step 5 and Step 6.

Random Pixel Sampling. According to Section III, a set of points on the target image I_t is randomly sampled. Since these points are used in calculating the appearance distances E_a for all poses during the process, the coordinates and the pixel values of the points are loaded into the uniform cache with LDU (load uniform) instructions in GPU, which benefits the appearance distance calculation by reducing the memory reading time.

Appearance Distance Calculation. The appearance distance $E_a(\mathbf{p})$ for each pose $\mathbf{p}$ in the set are calculated in parallel. For each pose $\mathbf{p}$, the system reads necessary information from memory and uniform cache to calculate $E_a(\mathbf{p})$ using (1) and (2). The result is written to the global memory. This task takes the longest processing time in the system because of the large amount of global memory reading for camera image pixel values. To improve the efficiency, we store the camera image pixel values in the texture memory, which benefit the system if the memory access patterns exhibit spatial locality. According to (3), the spatial distance between $\mathbf{u}_{i\mathbf{p}_1}$, $\mathbf{u}_{i\mathbf{p}_2}$ computed by two nearby poses $\mathbf{p}_1$, $\mathbf{p}_2$ is bounded, which makes the texture memory suitable for our system.

Pose Selection. In this part, we select the remaining poses $\mathcal{S}_L$ according to (4) parallelly. The memory is then re-allocated to make it compact.

Set Expansion. In theory, there will be 3 directions to expand ($[-\Delta, 0, \Delta]$) for each pose dimension, which results in 729 (3^6) directions for a pose. If $n_{\mathcal{S}_L}$ remaining poses are obtained in the previous stage, there are $729 n_{\mathcal{S}_L}$ poses in the expanded set. The number of expanded pose is too large for our system. Alternatively, we randomly generate 80 directions in parallel. The global memory usage for storing poses and appearance distances in the coarse-to-fine estimation is summarized in Fig 3(a).

C. Pose Tracker

For consecutively captured frames, pose tracking can be employed instead of pose estimation to accelerate the pose deriving task. We propose to take the pose estimated in the previous frame as the initial pose and perform tracking with the 3-scale search, where a 6D search pattern is employed. For the 6D pose search pattern, we assign five points in each of the rotation dimensions and seven points in each of the translation dimensions. Such unequal assignment is a practical consideration since the variation in translation motion is larger than that in rotation motion according to our statics.

There are two main reasons why we assign more than three points for each dimension. First, the overhead for passing parameters to GPU is hidden due to the larger number of parallel computing units. Second, the search range is larger for the proposed search pattern, which yields a faster search speed.

The goal for the pose search pattern is to find the pose with minimum E_a. The process includes the random pixel sampling and the appearance distance calculation, which are described in Section IV-B. After the pose with minimum E_a is obtained, we move the pattern to the position of the pose and diminish the pattern size to perform the search in the finer scale. A brief illustration in 2D view is shown in Figure 3(b). The blue pattern indicates the coarser scale, the green pattern indicates the finer scale, and the orange circle is the pose with minimum E_a found in the coarser scale. Considering the tradeoff between efficiency and accuracy, totally 3 scales are used in the pose tracker. In practical, the proposed system achieves 11 FPS for $640 * 480$ camera images.

V. Evaluation

Two experiments are conducted to evaluate the proposed system. In the first experiment, we use the synthetic dataset proposed in [10] to evaluate the proposed PEU and compare it with other methods. Second, we investigate the performance of the proposed PT using the real videos on a benchmark dataset. Finally, a comparison between the proposed system and other existing systems is conducted.

We measure the performance using rotation error and translation error and success rate. The rotation error $E_{\mathbf{R}}$ is defined

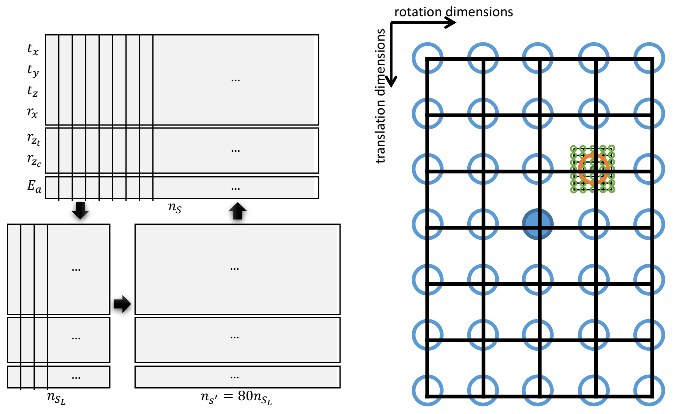


Fig. 3: (a) The global memory usage for storing poses and appearance distances in the coarse-to-fine estimation. The description below each memory block indicates the number of poses stored in the memory. (b) 2D view of the proposed 6D pose search pattern for the pose tracker.

as $E_{\mathbf{R}}(degree) = acosd((\mathrm{Tr}(\mathbf{R}^\top \cdot \hat{\mathbf{R}}) - 1)/2)$ while the translation error $E_{\mathbf{t}}$ is defined as $E_{\mathbf{t}}(\%) = \|\hat{\mathbf{t}} - \mathbf{t}\| / \|\hat{\mathbf{t}}\| \times 100$, where $\hat{\mathbf{R}}$ and $\hat{\mathbf{t}}$ is the ground truth rotation matrix and translation vector, respectively.

A. Synthetic dataset

The dataset consists of 8400 test images with 8 different target images. The target images are categorized as low texture, repetitive texture, normal texture and high texture. For each target image, there are normal condition and 20 different varying conditions.

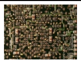
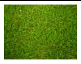
In this experiment, we compare PEU to APE described in Section III and ASIFT-based method. ASIFT-based method, which applies ASIFT [12], RANSAC [6], and OPnP [7] to estimate poses, has the upper-bound performance among feature-based systems. There are two evidences for such statement. First, to best of our knowledge, the ASIFT algorithm is the most powerful feature matching framework since it is affine invariant. However, it is not applied in any systems yet. Second, OPnP achieves state-of-the-art performance among PnP algorithms.

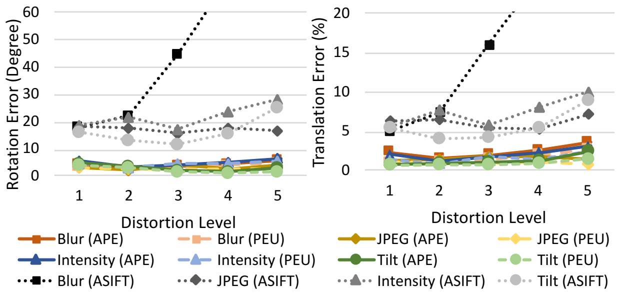
The results are shown in Table I and Fig 4(a). The proposed PEU and APE both show accurate and robust performance. ASIFT-based method, on the other hand, loses the effectiveness as the target image is textureless and the camera image is blurry. Note that the existing systems also fail to give correct estimation in the case of high tilt angle since the feature-matching methods they apply are not affine-invariant.

B. Real Videos

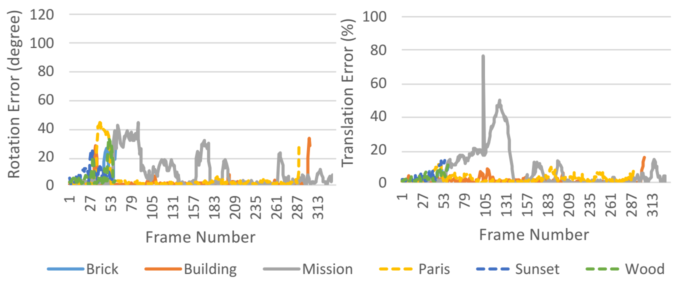
We use all "unconstrained" videos in the dataset by Gauglitz *et al.* [13] in this experiment. For each video, we use PEU to get the initial pose. After that, we use PT to track the poses until it loses the track. We record the pose sequences and compute the error in each video frame. Figure 4(b) shows the results. PT tracks the poses of targets Building, Mission, and Paris for more than 200 frames. As for the target Sunset and Wood, PT fails to track since the target images appear a flat color. The pose motion of target Brick in the video is extremely large, which is often out of the tracking range and

TABLE I: Evaluation results for the proposed PEU, APE, and ASIFT-based method in normal condition.

	Low Texture				Repetitive Texture				Normal Texture				High Texture			
																
	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$	$E_{\mathrm{R}}(°)$	$E_{\mathrm{t}}(\%)$
PEU	**1.37**	**0.53**	3.45	**0.67**	3.70	0.70	**5.53**	**1.04**	1.29	0.50	2.18	0.66	**1.59**	**1.07**	**1.50**	0.67
APE	1.82	0.55	**2.40**	1.01	3.07	0.84	5.56	2.91	**1.21**	0.61	2.31	0.69	2.51	2.16	3.14	1.26
ASIFT	72.1	24.3	5.07	0.74	**1.90**	**0.38**	6.37	2.59	2.08	**0.49**	**1.16**	**0.35**	51.2	16.7	2.76	**0.36**



(a)



(b)

Fig. 4: (a) Evaluation results for the proposed PEU, APE, and ASIFT-based method under blur, JPEG compression, intensity change, and tilt angle conditions with different degradation levels. (b) Experimental results by the proposed PT.

TABLE II: Comparison of the proposed system and state-of-the-art feature-based systems.

System	Core method	Platform	FPS	Resolution
Proposed	Direct method	NVIDIA Jetson TX1	11	$640 * 360$
Schaeferling [9]	SURF [3]	2 Xilinx Spartan-3E 2 ARM Cortex-A9	0.94	$640 * 480$
Rister [5]	SIFT	NVIDIA Tegra 250	7.9	$320 * 240$
Wang [4]	SIFT	Adreno320	5.9	$320 * 256$

results in losing the track. However, PT still tracks the poses over fifty frames which show the ability to track in severe conditions.

C. Comparison

A comparison of the proposed system and state-of-the-art feature-based systems are shown in Table II. Although Rister [5] and Wang [4] are built for feature detection and decription only, they can be integrated into 6 DoF pose estimation and tracking systems. The proposed system works at the fastest speed among these systems. According to the experiments described in Section V-A and V-B, the proposed system is capable of handling general planar targets. Moreover, it yields more robust performance than feature-based systems in cases that happen frequently in practical scenarios, which make it more ideal for real applications.

VI. Conclusion

In this paper, we propose D-PET, a direct 6 DoF pose estimation and tracking system implemented on an embedded GPU. The system consists of two main parts. The initial pose is estimated by the pose estimation unit, and the pose tracker applies the proposed 3-scale search to perform tracking. Compared to state-of-the-art feature-based systems, D-PET performs favorably regarding accuracy and robustness. The future work includes implementing the specific VLSI hardware to make D-PET available on wearable devices.

References

[1] H.-E. K. W.-Y. L. S.-H. K. K. C. J.-S. P. Jae-Sung Yoon, Jeonghyun Kim and L.-S. Kim, “A unified graphics and vision processor with a 0.89 w/fps pose estimation engine for augmented reality,” *IEEE Transactions on Very Large Scale Integration Systems*, 2013.

[2] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, 2004.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf),” *Computer Vision and Image Understanding*, vol. 110, no. 3, 2008.

[4] G. Wang, B. Rister, and J. R. Cavallaro, “Workload analysis and efficient opencl-based implementation of sift algorithm on a smartphone,” in *IEEE Global Conference on Signal and Information Processing*, 2013.

[5] B. Rister, G. Wang, M. Wu, and J. R. Cavallaro, “A fast and efficient sift detector using the mobile gpu,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

[6] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[7] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, “Revisiting the PnP Problem: A Fast, General and Optimal Solution,” in *IEEE International Conference on Computer Vision*, 2013.

[8] G. Simon, A. W. Fitzgibbon, and A. Zisserman, “Markerless tracking using planar structures in the scene,” in *IEEE International Symposium on Mixed and Augmented Reality*, 2000.

[9] M. Schaeferling, U. Hornung, and G. Kiefer, “Object recognition and pose estimation on embedded hardware: Surf-based system designs accelerated by fpga logic,” *International Journal of Reconfigurable Computing*, vol. 2012, p. 6, 2012.

[10] H.-Y. Tseng, P.-C. Wu, M.-H. Yang, and S.-Y. Chien, “Direct 3d pose estimation of a planar target,” in *IEEE Winter Conference on Applications of Computer Vision*, 2016.

[11] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[12] G. Yu and J.-M. Morel, “Asift: A new framework for fully affine invariant image comparison,” *Image Processing On Line*, 2011.

[13] S. Gauglitz, T. Höllerer, and M. Turk, “Evaluation of interest point detectors and feature descriptors for visual tracking,” *International Journal of Computer Vision*, vol. 94, no. 3, pp. 335–360, 2011.