A PC-Cluster based Multi-Camera System for Real-Time Multiple Objects Tracking

Bamrungthai P.* and Sangveraphunsiri V.

Department of Mechanical Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand

Abstract

This paper presents a PC-cluster based multi-camera system for real-time multiple objects tracking. The PC-cluster, a group of linked computers, is developed by using PCI-to-PCI data mover card to improve real-time processing performance and provide system scalability. Five processing modules have been designed and implemented to control system operations. Target objects are in spherical shape. Four IEEE-1394a cameras with hardwired synchronization are used for system testing. After identifying correspondence objects in different views, 3-D coordinates of each target have been reconstructed. Experimental results show that the system can identify and track multiple objects moving in 3-D space correctly. The maximum frame rate for tracking is 12 frames per second. In the near future, the system can be applied for specific application when additional processing modules are added to the system.

***Keywords**: Multi-camera system, PC-cluster, Multiple objects tracking*

1 Introduction

Object tracking using vision technology is an important step in many applications such as motion capture, unmanned aerial vehicle (UAV) target tracking and automatic surveillance system. Because of its versatile applications, tracking system has become an active research area.

Several factors such as object feature, object complexity and characteristic of motion affect system configuration and applied method. In case of 3-D tracking, stereo vision [1, 2], two cameras system, has widely used to make 3-D reconstruction possible. But its main drawbacks are limited field of view and object occlusion. If object feature is not available for one or both of cameras, 3-D reconstruction cannot be performed. This is one of failure causes in tracking process. To solve for such problem, multiple cameras have been used to increase viewable and decrease occlusion.

A large amount of data obtained from multiple cameras needs to be processed by processing unit of the system. This affects real-time tracking ability. PC-cluster, a group of linked computers, has been adopted to apply for multiple cameras system [3, 4] to improve real-time processing performance. Typically, system operations have been separated into parts to reduce computational load on each computer. Synchronization system has been applied for communication between computers.

In this research, a real-time multiple objects tracking system using multiple cameras has been developed on a PC-cluster which is constructed with PCI-to-PCI data mover card. Processing modules have been developed for system operations control in multiple objects tracking task. Target objects are spherical objects with almost identical diameter. In many applications, such object has been used as marker to be tracked to reduce complexity of object detection.

In this case, object identification is required for 3-D reconstruction and tracking process.

2 The PC-cluster

The developed system consists of hardware and software components. Hardware includes computer system and devices used for communication between computers and also cameras and their accessories, e.g. camera cables. Software includes application software for system operations control in each computer. Within application software, processing modules are embedded.

The PC-cluster developed in this research consists of two types of computers. One is main or local computer for overall system operations control such as camera settings, parameters adjustment for camera calibration and object tracking, getting user command, and graphical output display on monitor. The other is remote computer used for image capturing, object detection and object identification and tracking from attached cameras. Application software is contained in each computer according to type of computer as shown in the next section. Configuration of the PC-cluster is shown in figure 1.

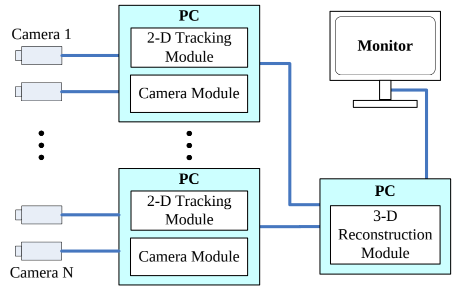

Figure 1: Configuration of the PC-cluster

Because of system architecture design [4], only one computer is allowed to be main or local computer. It controls operations which are contained in it and in all remote computers. More than one computer is allowed to be remote computer in the system.

3 System Implementation

In this paper, two computers with Microsoft Windows XP operating system have been used for system implementation. Application software in the local computer has been designed as 'control application' that has graphical user interface (GUI) to receive user input and display tracking results. But software in remote computer has been designed as 'service application' that runs in the background of Windows operating system and has no user interface to reduce computational load in remote computer.

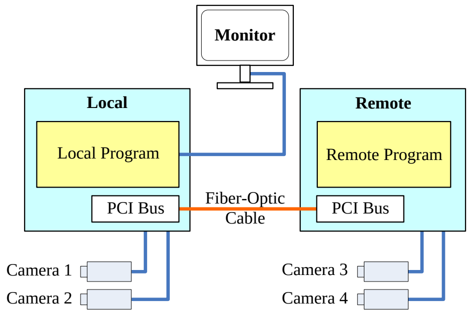

Figure 2: The system schematic diagram

The system implementation schematic diagram has been shown in figure 2. The two computers have been connected by using PCI-to-PCI data mover card with fiber-optic connection. Two cameras connect to each computer via IEEE-1394a interface.

From software design, tasks for tracking multiple objects have been separated into modules as follows:

1) Camera module
2) 2-D tracking module
3) Cluster module
4) Calibration module
5) 3-D reconstruction module

The arrangement of processing modules in each computer has been shown in figure 3.

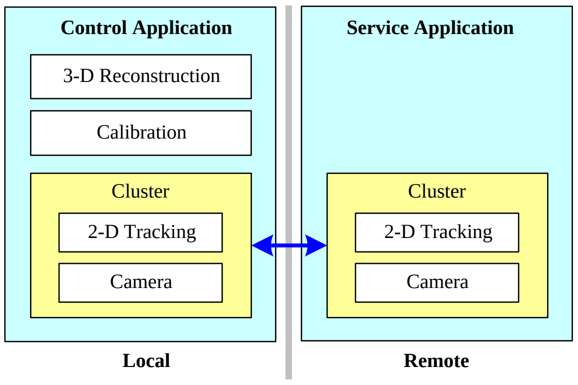

Figure 3: Processing modules arrangement inside the application software

4 3-D Tracking process

Before starting 3-D tracking process, all cameras need to be calibrated to obtain camera parameters for using in 3-D reconstruction. The camera calibration methods developed by Tsai [6] and Zhang [7] have been used in this research. User can select method from control application in the main computer. Details of the camera calibration can be found in [4, 5]. Camera synchronization has been done by using hardwired trigger which one camera has been chosen as primary camera to receive trigger signal from the application software and send hardware trigger signal to all secondary cameras to ensure minimum time delay of image capture.

Processing steps for 3-D tracking, after camera calibration, can be shown in figure 4.

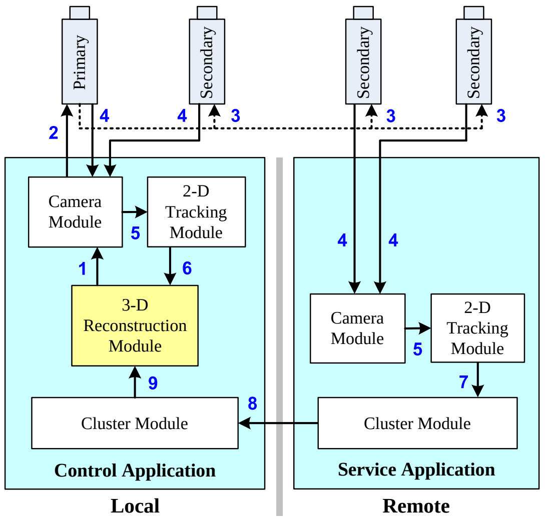

Figure 4: Processing steps for 3-D tracking

Processing steps in figure 4 can be described as follows:

1) 3-D reconstruction module sends command to Camera module to capture images.
2) Camera module sends capturing signal only to primary camera.
3) Primary camera sends hardware trigger signal to all secondary cameras.
4) All cameras capture images and send them to corresponding Camera module.
5) In each computer, Camera module sends images to 2-D tracking module to detect and identify objects.
6) For local computer, 2-D tracking module sends object positions to 3-D reconstruction module directly.
7) For remote computer, 2-D tracking module sends object positions to cluster module.
8) Cluster module transforms object positions into message and sends it to cluster module in local computer.
9) Cluster module in local computer receives message, converts it back to object positions, and sends object positions to 3-D reconstruction module.
10) 3-D reconstruction module identifies correspondence objects in different cameras and calculates 3-D positions for each object.

5 Object tracking

For tracking multiple objects, object positions in image are detected and estimated for identifying and tracking individual object. Predicted object position will be called 'estimated objects' in this paper. Estimated objects with constant acceleration motion assumption during a short time interval will be associated with detected objects. Object identification and tracking process has been shown in figure 5.

In this paper, the algorithm used for identification and tracking has been adopted from [8] which formulated the problem as finding a combination of estimated objects $(E_1, E_2, \dots, E_{n_d})$ that maximizes the reward $J = \sum_{i=1}^{n_d} \rho(D_i, E_i)$, where n_d is the number of detected object, D_i is the i-th detected object, $E_i \in \{\phi\ \varepsilon\}$ is the estimated objects associated with D_i, ε is the set of all estimated objects, and ϕ is a null object.

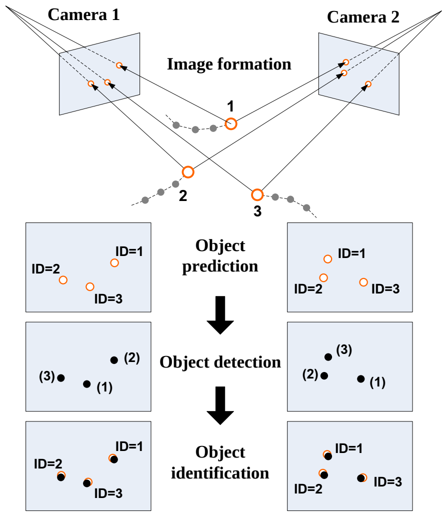

Figure 5: Object identification and tracking process

If node between detected objects D_i and estimated objects $E_j \in \{\phi\ \varepsilon\}$ is written as (D_i, E_j). The algorithm can be outlined as follows:

1) Create the initial node (ϕ, ϕ) with the total reward $\overline{\rho}(\phi, \phi) = 0$.
2) Find the unopened node (D_i, E_j) with the largest total reward.
3) For all $E_m \in \{\phi\ \varepsilon\}$, create new nodes (D_{i+1}, E_m) as children of (D_i, E_j) and mark node (D_i, E_j) as opened. If $E_m \in \{\varepsilon\}$ and E_m is already used in the ancestors of (D_i, E_j), the object is ignored.

4) For each new node, compute its total reward by

$$\bar{\rho}(D_{i+1}, E_m) = \bar{\rho}(D_i, E_j) + \rho(D_{i+1}, E_m) \qquad (1)$$

where $\rho(D_i, E_j)$ is the reward of the pair (D_i, E_j) and $\bar{\rho}(D_i, E_j)$ is the total reward of node (D_i, E_j).

5) If $i+1 = n_d$, mark the new nodes as goal and opened.
6) If there is no more unopened node or the number of goals reaches the predefined maximum, proceed to 7). Otherwise return to 2).
7) Find the goal node with the largest total reward and obtain the tracking result by tracing up its ancestors.

If $E_j \neq \phi$, the reward $\rho(D_i, E_j)$ is defined as a function of the distance between the detected and estimated objects:

$$\rho(D_i, E_j) = \frac{C}{d(D_i, E_j) + C} \qquad (2)$$

where $d(D_i, E_j)$ denotes the distance between the detected object D_i and estimated object E_j, and C is a positive constant. The reward for $E_j = \phi$ is defined as

$$\rho(D_i, \phi) = \frac{C}{d_{\max} + C} \qquad (3)$$

where $d_{\max}$ is the predefined maximum distance between a detected object and its estimated position. In implementation, node that has distance $d(D_i, E_j)$ larger than $d_{\max}$ is not added to save memory and search time.

6 3-D Reconstruction

After identifying objects, the next step is to reconstruct 3-D position of each object. Using corresponding object feature position in images in different views, 3-D coordinates of each object can be found as shown in figure 6. Multiple views reconstruction has been done in the final steps of the reconstruction process. The linear triangulation method with least square [10] has been applied for the implementation system.

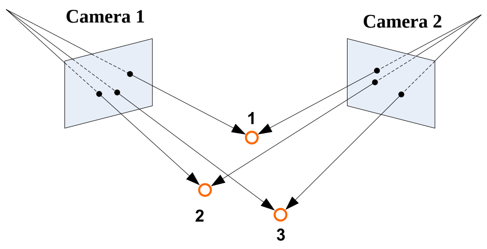

Figure 6: 3-D reconstruction of multiple objects in case of two cameras

The reconstruction problem is formulated as finding combinations of detected objects and an estimated object $(M^0_{i_0}, M^1_{i_1}, M^2_{i_2}, \ldots, M^{n_c}_{i_{n_c}})$ that have large total reward

$$J = \rho(M^0_{i_0}, M^1_{i_1}, M^2_{i_2}, \ldots, M^{n_c}_{i_{n_c}}) \qquad (4)$$

where

n_c is the number of cameras,

$M^0_{i_0} \in \{\phi\ \varepsilon\}$, $M^j_{i_j} \in \{\phi\ D^j\}$ $(j > 0)$, and

D^j is the set of object detected at camera j.

$M^0_{i_0} = \phi$ indicates that the object is a new object and not associated with any estimated object, and $M^j_{i_j} = \phi$ $(j > 0)$ indicates that the object is not visible from camera j.

The method used in [8] has been modified and applied in this research for 3-D reconstruction which is outlined as follows:

1) Create the initial nodes $M^0_i \in \{\phi\ \varepsilon\}$ with zero reward.
2) Find the unopened node M^j_i with the largest reward and mark node M^j_i as opened.
3) Create children nodes of M^j_i as follows:
   a. If the initial node $M^0_{i_0}$ of M^j_i is in ε and an object $M^{j+1}_k \in D^{j+1}$ has the same ID as $M^0_{i_0}$, add a child node M^{j+1}_k. If no marker in D^{j+1} has the same ID, add a child node as ϕ.
   b. Otherwise add children nodes for all $M^{j+1}_k \in \{\phi\ D^{j+1}\}$.
4) For each child node, compute its reward as follows:
   a. Compute the distance between the vector corresponding to marker M^{j+1}_k and the current intersection $p(M^j_i)$. Computation of $p(M^j_i)$ will be discussed later. If $M^{j+1}_k = \phi$, use the predefined maximum distance $d_{\max}$.

   b. Compute the local reward $\rho(M_k^{j+1})$ based on the distance and compute the total reward $\overline{\rho}(M_k^{j+1}) = \rho(M_k^{j+1}) + \overline{\rho}(M_i^{j})$.

5) If $i+1=n_c$, mark the new nodes as goal and opened.
6) If there is no unopened node or the number of goals reaches the predefined maximum, proceed to 7). Otherwise return to 2).
7) Gather goal nodes whose average distance from the estimated object position is smaller than user-defined threshold and with more than $n_{\min}$ valid markers.
8) Recalculated reconstruction positions of each object using linear triangulation method [9] using correspondence object positions obtained from step 7) to get better accuracy.

The intersection of a pair of vectors is defined as the center of nearest points. Current intersection $p(M_i^j)$ of node M_i^j is defined as the average of intersections of all vector pairs.

7 Experimental results

7.1 *Experimental setup*

The developed system with experimental setup for accuracy testing has been shown in figure 7. Computers used for experiments include local computer which uses Intel Core 2 Quad processor at 2.83 GHz and remote computer with Intel Pentium 4 processor at 3.40 GHz. Image size, in pixel, used for processing is 640x480. Cameras used in this research are PixeLINK PL-A741-BL CMOS camera with 8-mm lens.

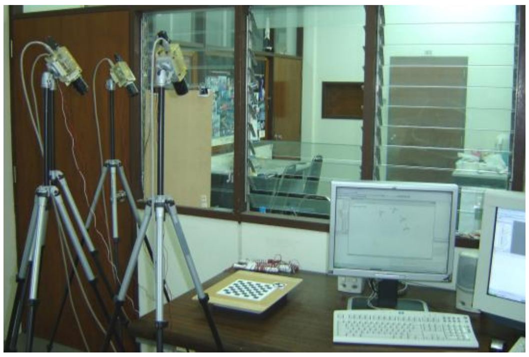

Figure 7: The system setup for accuracy testing

The first experiment is accuracy testing for 3-D reconstruction of known points (corners of calibration pattern). The 7x7 corner points have been detected and reconstructed by using linear triangulation method. The exact 3-D positions and reconstruction positions are used for calculating 3-D position error. Simulation for occlusion has been done by disable some cameras on reconstruction. The application software for local computer with images for the testing has been shown in figure 8.

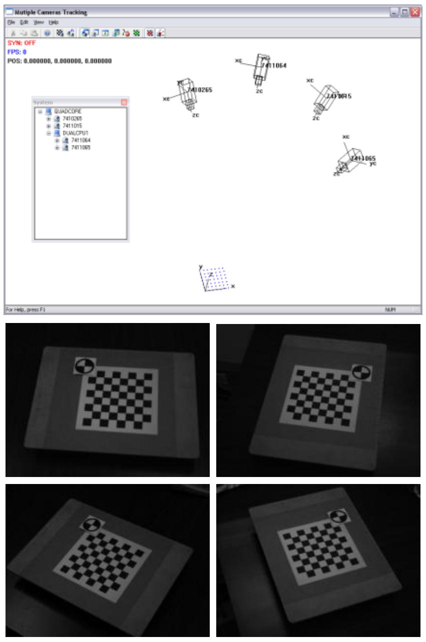

Figure 8: The control application with images of calibration pattern from four cameras

The developed system can detect and track a moving target, a spherical ball, attached to Mitsubishi PA10-7C manipulator arm as shown in figure 9. Single object tracking has been performed in the second experiment.

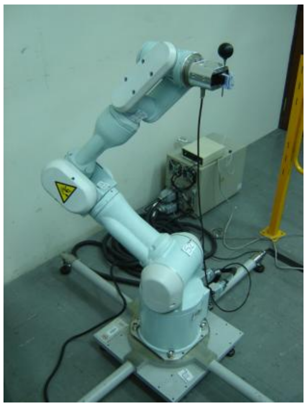

Figure 9: PA10-7C manipulator arm with a spherical target

Table 1: Reconstruction error in accuracy testing

No. of cameras (Camera serial number)	Average error (mm)	Standard deviation (mm)	Maximum error (mm)
4	0.0831	0.0335	0.1810
3 (w/o 7410265)	0.1097	0.0748	0.3290
3 (w/o 7411015)	0.0934	0.0370	0.2000
3 (w/o 7411064)	0.0892	0.0369	0.1990
3 (w/o 7411065)	0.0747	0.0339	0.1660
2 (w 7410265, 7411015)	0.1131	0.0636	0.3010
2 (w 7410265, 7411064)	0.1922	0.1606	0.6580
2 (w 7410265, 7411065)	0.1031	0.0594	0.3030
2 (w 7411015, 7411064)	0.1051	0.0483	0.2350
2 (w 7411064, 7411065)	0.0816	0.0292	0.1490

w = with, w/o = without

In the third experiment, PA10-7C manipulator arm has been used for moving target objects which are three spherical objects with 20 mm diameter. The objects are arranged in straight line with 50-mm spacing between objects.

7.2 *Results*

The accuracy testing result is shown in table 1. It shows the average, standard deviation, and maximum errors obtained from the experiments. The results show that reconstruction error is not exceeded 1 mm.

Results from single target tracking have been shown in figure 10-14. Fig. 10 shows tracking of the target object with s-curve velocity profile. The robot arm is program to move in y-direction with acceleration and deceleration set at 20 mm/s^2. The total distance is 300 mm with 40 mm/s constant velocity. In figure 11 and 12, results from the measurements and the robot arm positions have been compared when robot moves in planar circular motion in XY and YZ plane, respectively. The robot moves in circle with 100-mm radius on the specified plane with angular velocity 0.05 rad/sec. In figure 13 and 14, results from the measurements and the robot arm positions have been compared when robot moves in helical motion with helix's axis are in Z and X direction, respectively. The robot moves in helical motion with 100-mm radius and distance along axis of helix 200 mm with angular velocity 0.05 rad/sec and linear velocity along helix axis 2.5 mm/sec. Dash lines in the figures are detected points obtained from the system and solid lines are object points generated by PA10. The results show that the system can track robot motion with acceptable accuracy in both position and velocity.

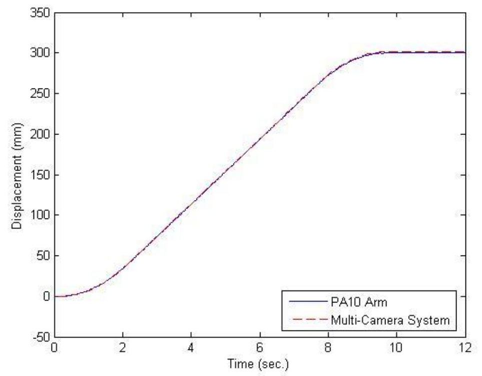

Figure 10: Linear motion with s-curve velocity profile

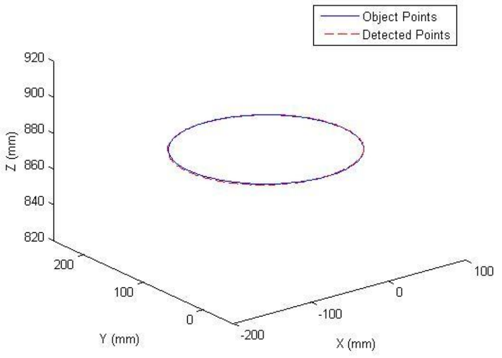

Figure 11: Single target moves in XY plane

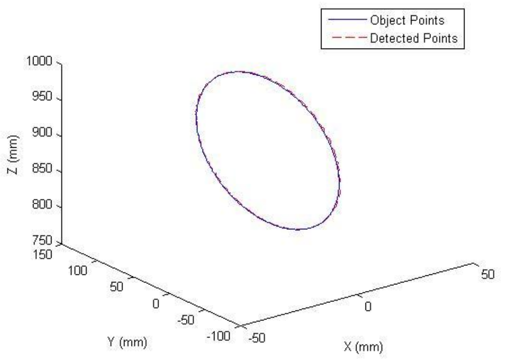

Figure 12: Single target moves in YZ plane

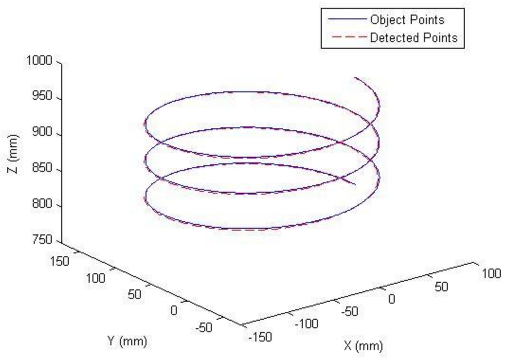

Figure 13: Single target moves in helical motion in Z-direction

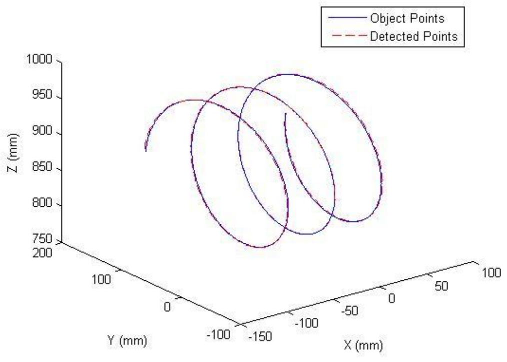

Figure 14: Single target moves in helical motion in X-direction

For the third experiments, results have been shown in figure 15-18. In figure 15 and 16, results from the measurements and the robot arm positions have been compared when robot moves in planar circular motion in YZ and XY plane, respectively with same motion in the second experiments for single object tracking. In figure 17 and 18, results from the measurements and the robot arm positions have been compared when robot moves in helical motion with helix's axis are in X and Z direction, respectively. The robot moves in helical motion with the same motion with the single target case. Solid lines in the figures are detected points obtained from the system and dash lines are object points generated by PA10.

The results show that the system can identify and track objects correctly at maximum frame rate 12 frames per second. A well prepared environment such as using active target can be used to reduce exposure time of the cameras. Tracking frequency can be improved by using faster cameras. Fine tuning application software's implementation is also affect system performance.

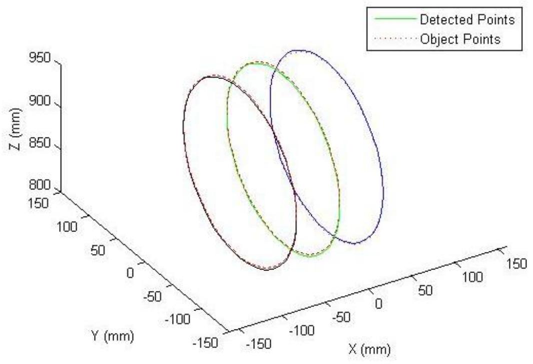

Figure 15: Planar circular motion in YZ plane

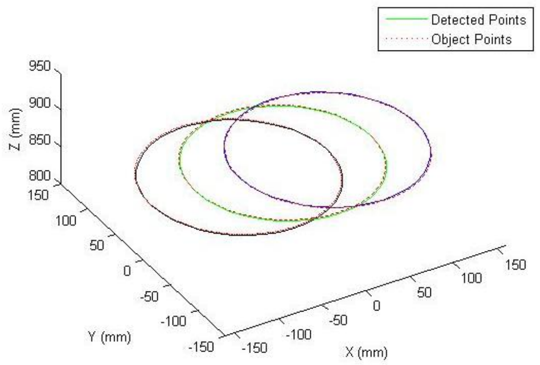

Figure 16: Planar circular motion in XY plane

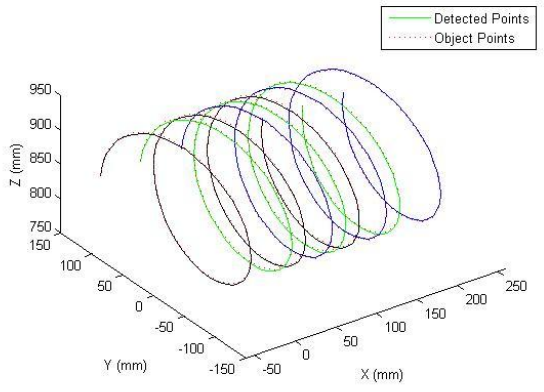

Figure 17: Helical motion with axis of helix in X axis

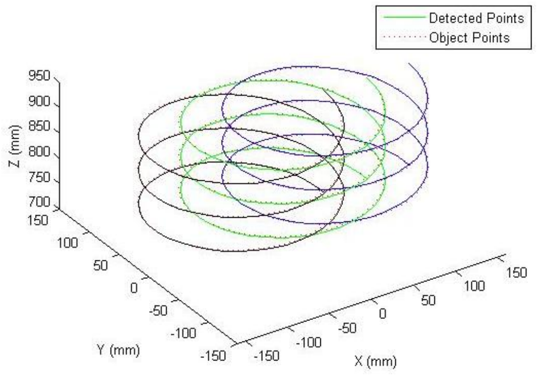

Figure 18: Helical motion with axis of helix in Z axis

8 Conclusions

This paper presents the development of a real-time multiple objects tracking system using multiple cameras and a PC-cluster. PCI-to-PCI data mover card has been used for data transfer between computers. Application software with processing modules has been designed for system operations control. Experimental results show that the system can track spherical objects moving in 3-D space correctly. For future development, additional processing modules can be added to the system to apply for specific task.

Acknowledgements

This work has been supported by Chulalongkorn University under Chulalongkorn University Centenary Academic Development Project.

References

[1] Trucco, E. and Verri, A., 1998. *Introductory techniques for 3-D computer vision*. New Jersey, USA: Prentice-Hall.

[2] Sangveraphunsiri, V. and Chooprasird, K, 2009. A 3D Particle Tracking System using Stereo Vision. *The 23rd Conference of Mechanical Engineering Network of Thailand*, Chiang Mai.

[3] Arita, D., Yonemoto, S., and Taniguchi, R., 2000. Real-time computer vision on PC-cluster and its application to real-time motion capture. *5th IEEE International Workshop on Computer Architectures for Machine Perception*: 205-214.

[4] Uttamang, K., and Sangveraphunsiri, V., 2007. Multi-camera system using PC-cluster for real-time 3-D pose estimation. *Thammasat Int. J. Sc. Tech.* 12 (April-June 2007): 52-63.

[5] Sangveraphunsiri, V., Uttamang, K. and Pedpunsri, P., 2009. A Multi-Camera System on PC-Cluster for Real-time 3-D Tracking. *The 23rd Conference of Mechanical Engineering Network of Thailand*, Chiang Mai.

[6] Tsai, R., 1987. A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology using Off-The-Shelf TV Cameras and Lenses, *IEEE Journal of Robotics and Automation*, vol. 3. no.4, pp. 323-344.

[7] Zhang, Z., 2000. A Flexible New Technique for Camera Calibration, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22.

[8] Yamane, K., Kuroda, T, and Nakamura, Y., 2004. High-Precision and High-Speed Motion Capture by Combining Hetegeneous Cameras, *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Japan.

[9] Hartley, R., and Zisserman, A., 2003. *Multiple view geometry in computer vision*. 2nd ed. Cambridge, UK: Cambridge University Press.

[10] Hartley, R., and Sturm, P., 1997. Triangulation, *Computer Vision and Image Understanding*, Vol. 68. No. 2, November, pp. 146-157.