The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Mobile Robotic Dynamic Tracking for Assembly Tasks

Bradley Hamner, Seth Koterba, Jane Shi, Reid Simmons, Sanjiv Singh

Abstract—**Traditional industrial robots have been widely used in automotive manufacturing for nearly 30 years. However, there have been very few attempts to automate mobile robotic systems for final assembly operations, despite their potential for high flexibility and capability. This paper focuses on methods of tracking a dynamic moving vehicle that is similar to the vehicle body on a moving assembly line. We have investigated two tracking methods, one using a laser scanner and the other using a visual fiducial marker. We have also studied the tracking performance of a mobile base using the pure pursuit algorithm with low pass filtering. Experimental results are presented to illustrate the remaining main challenges in achieving robotic assembly on moving assembly lines.**

I. Introduction

Automotive general assembly tasks have been performed by human operators on moving assembly lines since the early 1930s. Typically, thousands of assembly operations are divided into hundreds of workstations along moving assembly lines. The operators follow the vehicle body movement within a workstation footprint to assemble a variety of parts onto the vehicle. In order to achieve robotic assembly on moving assembly lines, a *line tracking* system, in which the partially assembled vehicle's movement is tracked to its highest possible fidelity, must be developed. If the robot can accurately track the assembly line movement, the assembly task can be performed as if there were no relative motion between the robot and the vehicle body.

Line tracking by traditional industrial robots has been studied by DeSouza [1]. In collaboration with NIST, DeSouza's line tracking performance has been explored in a preliminary study [2]. In addition, General Motors Corporation has recently investigated assembly conveyor dynamic motion [3] and evaluated the performance of current state-of-the-art commercial line tracking solutions by traditional industrial robots [4].

Mobile manipulators, such as shown in Figure 1, offer high mobility and manipulability. An ideal utilization of the motion redundancy in the mobile manipulator is to perform assembly tasks on a moving vehicle body while tracking. The simplest partition is to separate the tracking and assembly motions, so that the two tasks can be performed by independent control loops. In previous work, we have shown that a manipulator arm using force control can perform fine-control assembly tasks (less than 2 mm error) in a static scenario [5].

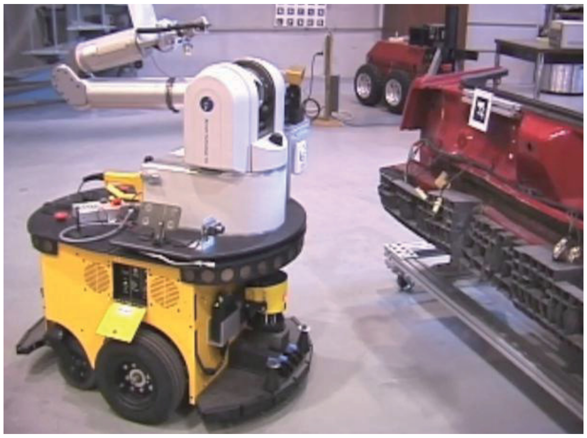


Figure 1. Our mobile manipulator consists of a PowerBot mobile base with a 7-DOF WAM arm, a Bumblebee stereo camera mounted on a pan-tilt unit, and a SICK laser scanner. Also shown is the task board with a visual fiducial marker used for pose estimation.

Two main approaches – integrated and decoupled – exist in coordinating mobility and manipulation in mobile manipulators. The integrated approach has been investigated by several researchers. Khatib [6] established a fundamental framework for dynamics-based end-effector task control and platform posture control. Tan and colleagues [7, 8] demonstrated the nonlinear control feedback advantages using unified dynamic models for mobile manipulators subject to nonholonomic constraints. Other coordinated control techniques include maintaining preferred configurations measured by manipulability [9] and formulating the differential motions that are feasible with respect to the nonholonomic constraints [10]. The decoupled approach is preferred when the locomotion is imprecise, as demonstrated by Shin [11]. Our approach, outlined in this paper, explores the decoupled approach where the platform control is used for the task of mobile tracking. If the mobile base could be proven to track a moving assembly line adequately, with small relative tracking error, assembly on moving assembly lines could be achieved by independently combining the mobile tracking and robot arm assembly manipulation.

This paper presents the investigation and development of techniques for a mobile robot to dynamically track the movement of a moving task board that emulates a moving

This work was supported by General Motors Corporation Manufacturing Systems Research Laboratory at R&D Center, Warren Michigan.

Bradley Hamner, Seth Koterba, Reid Simmons, Sanjiv Singh are with Robotics Institute at Carnegie Mellon Institute, Pittsburgh, PA 15213 USA

Jane Shi is with Manufacturing Systems Research Lab at GM R&D Center, Warren, MI 48090 USA (586-986-0353; fax: 586-986-0574; e-mail: Jane.Shi@gm.com).

978-1-4244-3804-4/09/$25.00 ©2009 IEEE

vehicle body on an assembly line. Two tracking methods are presented. The first one uses a visual fiducial marker on the moving task board. The second one uses a laser scanner mounted on the mobile robot. We also present a pure-pursuit algorithm that has been utilized to control the linear and angular velocities of the mobile base. Finally, we present experimental results for the mobile manipulator's performance in a dynamic tracking scenario. The results clearly demonstrate the remaining challenges in order for the robot to perform assembly tasks on a moving task board.

II. Visual Pose Estimation Algorithm for Tracking

A visual pose estimation algorithm, utilizing an ARTag fiducial [15], produces a six dimensional relative transform between the mobile robot and the fiducial marker attached to the task board (see Figure 1). In order to track a moving vehicle body that moves at 30mm/sec with a peak acceleration that could cause instantaneous speed change of 30mm/sec within one second, the visual pose estimation algorithm must produce pose outputs at a minimum rate of 10Hz. As with many vision-based pose estimation algorithms, the primary bottleneck in our pose estimation process is the fiducial detection step. To improve the pose update rate from less than 3Hz to more than 10Hz, we chose to reduce the amount of data that the closed-source ARTag library needs to process. Using wide field of view Bumblebee [16] cameras and operating at typical distances of more than 1m, only a small subset of each image contains useful fiducial data. An initial estimation of which portion of the image contains the fiducial object of interest can be used to crop the image at the right location. From initial system start up, until a fiducial is first detected, the system must perform full image processing. However, on subsequent iterations, the previous position of the fiducial, in image space, is used to crop the full image to a smaller sub-image.

Due to a constraint of the ARTag software, processing continuously variable sub-image sizes is not feasible. Therefore, we chose to use a fixed set of three sub-image sizes. The full image size of the Bumblebee camera used on the mobile robot is 1024 X 768 pixels and the sub-image window size options chosen were 1/3, 1/5, and 1/7 in each dimension. Thus, 1/9, 1/25, and 1/49 of the full image area (342 x 256, 205 x 152, 146 x 110 pixels) are processed by the fiducial detection step (Figure 2, left). Experiments have demonstrated that the three sub-image sizes are sufficient for entire working range of the mobile robot. The 10mm square fiducial object occupies less than 1/50 of the full image at 2 meters and less than 1/9 of the full image at the closest assembly distance of 0.5 meters. Thus the 1/7 size option is sufficiently large for long distance, while the 1/3 size option is useful for the closest working distance.

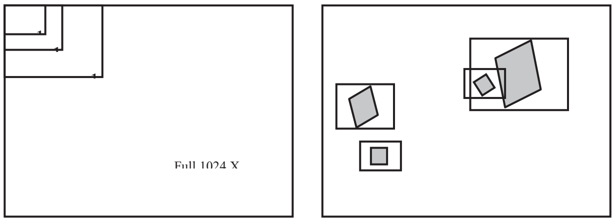


Figure 2. Left: Three sub-image sizes, 1/3, 1/5, 1/7, are chosen
Right: Found fiducials are shown within a sub-image

Choosing which of these three sub-window options to use during each iteration of processing is a dynamic process based on the size of the fiducial in the previous image. The fiducial corners, in image space, are stored from the previous iteration; therefore, the area of the previously found fiducial can be computed. To accommodate any change in the fiducial position between two subsequent iterations, the area is multiplied by a factor of 1.5, chosen empirically, and then compared with the 1/3, 1/5 and 1/7 sub-image sizes. The sub-image size that completely encompasses the expanded fiducial size is selected and centered at the fiducial centroid position from the previous iteration.

If the motion between two subsequent iterations is large enough to cause the fiducial to move outside the selected sub-image area, or the fiducial is occluded, the fiducial detection step will fail. In this case, the next larger sub-image is selected and the fiducial detection step is repeated with the next sub-image. This process is repeated until a valid fiducial is detected or the full image has been processed. The maximum number of repeats at the fiducial detection step is four, in which case about 120% of the full image pixels will be processed. The pose estimation outputs will be slowed down to 3Hz in this worst case scenario.

When multiple fiducials are detected at the previous iteration, the fiducial detection step is performed for each of the found fiducials with its own sub-image area. It is possible that multiple fiducials occur within one sub-image area. It is also possible that one fiducial appears in multiple sub-image areas (Figure 2, right). Therefore it is necessary to track which fiducials have already been found during a given iteration, to eliminate the problem of redundant detection of fiducials. The overall effect of this speedup process is an increase of speed up to the grab frequency of the cameras, which is 20Hz.

If a new fiducial moves into the view of the cameras, it will not necessarily be detected in any of the subimages. To assist discovery of new fiducials, a full image processing is needed. Every 40 iterations (approximately every 2 seconds), a full image search is performed. The complete algorithm is illustrated in Figure 3.

III. Pose Estimation with Laser Measurement Data

Tracking with a laser scanner requires a physical object that can be modeled easily and detected by the scanner. A "laser fence" was added to the task board at the height of the laser scanner (a Sick S300 [17]) that is mounted on the front of the mobile base (Figure 1). The fence is linear with a distinctive, protruding post at each end, and is modeled with a set of points spaced 5 mm apart as outlined by the blue

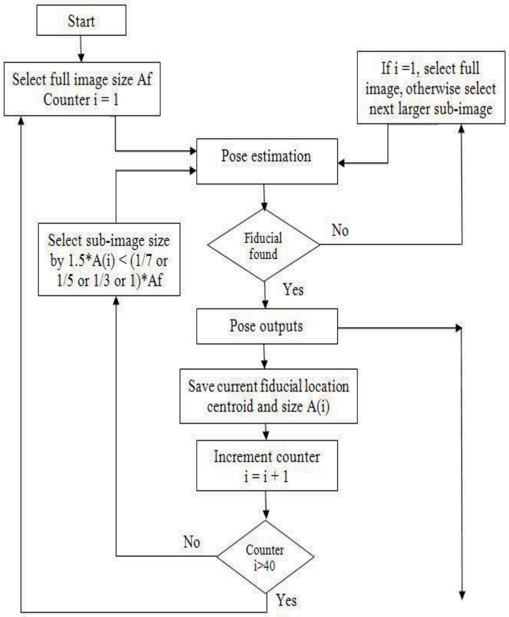

Start
Select full image size Af
Counter i = 1
If i =1, select full image, otherwise select next larger sub-image
Pose estimation
Select sub-image size by 1.5*A(i) < (1/7 or 1/5 or 1/3 or 1)*Af
Fiducial found
No
Yes
Pose outputs
Save current fiducial location centroid and size A(i)
Increment counter i = i + 1
No
Counter i>40
Yes

Figure 3. Visual fiducial pose estimation algorithm for tracking

point set in Figure 6.

Given the laser measurement data and the object model, a three dimensional relative pose can be estimated. The raw laser data is collected as a series of range measurements. The laser scanner has a 270 degree field of view, consisting of 540 individual measurements, and can detect objects at up to 30 meters away. Using the known minimum and maximum angles of the scan range and the angular offset between each individual range measurement, the raw range data are converted into point data in the Cartesian coordinate frame that is attached to the mobile robot base.

The Iterative Closest Point (ICP) [13] algorithm, commonly used for matching two point clouds, is used to match the laser measurement data and the linear model. Initial tests revealed that it has severe limitations. First, depending on the initial pose values, ICP can find a local minimum of the root mean square (RMS) error, leading to incorrect results. In particular, the initial pose values have to be within 10 degree and 50mm linear distance from the true pose for ICP to converge reliably to the actual pose. Second, the full range of data cannot be used to match with the model, since this would introduce many more local minima. Only the points corresponding to the laser fence itself should be used, and all other points should be segmented out of the entire point cloud. To overcome the above problems, effective initial preprocessing of the laser measurement data, prior to using ICP, is required.

The goal of preprocessing is to identify the laser measurement points that correspond to the laser fence and get an initial, pre-ICP, estimate of its pose. A two step process is used to identify and group the laser measurement data into independent clusters (segmentation step). The algorithm takes advantage of the linearity of the laser fence on the task board. The first step is to use the Random Sample Consensus (RANSAC) algorithm [14] to identify the strongest linear feature in the data (we assume that there are no walls or linear features in the data that would have a stronger response than the laser fence). Data points are searched sequentially, starting with the point corresponding to the counterclockwise-most scan. Taking this point and a second point, generated from the range measurement N (chosen to be 40) angular segments ahead in the clockwise direction, a line is generated. Next, the perpendicular distance of all remaining points to this line is computed. If this distance is within a prescribed threshold (currently 150mm), the point is counted as a hit on that line. By sequentially stepping through the laser data by increments of M (5) scans, a series of lines are generated. The line that receives the most hits is considered to be a line roughly aligned with the task board. Figure 4 shows two examples of RANSAC fitting results. The point data around the line C is a better candidate for the model matching.

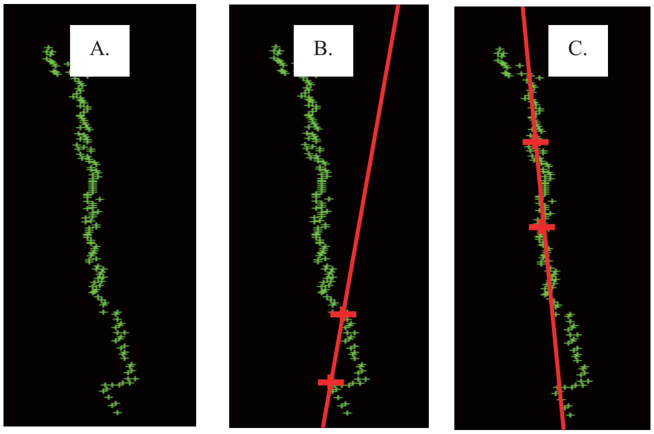


Figure 4. A) Raw laser data of the task board. B) and C) two examples of RANSAC fitting. Line C has more hits than line B, and so it is more likely associated with the task board's laser fence.

The second step is to cluster all the laser measurement data that belong to the task board. Based on the previous step, a rough line through the task board has been identified that is the best candidate among all generated lines. A middle point can be found that is half way between the two points that were used to generate the candidate line. Starting from the middle point, in each direction, subsequent points are checked if they are within a distance threshold of the previous point (currently 100mm). If a point does not meet this threshold, at least two additional points past the first out-of-threshold point are checked to ensure that all task board points are found so that one bad measurement point would not cause the clustering process to stop. After three consecutive points that are outside the defined threshold, the clustering algorithm stops and assumes that all points that correspond to the task board are found in both directions. In practice, this clustering algorithm is robust because of the properly selected threshold. Figure 5 shows an example of clustering results that have identified all laser measurement data for the task board among nearby objects.

After the laser data associated with the fence have been identified, the next step is to roughly align the cluster with

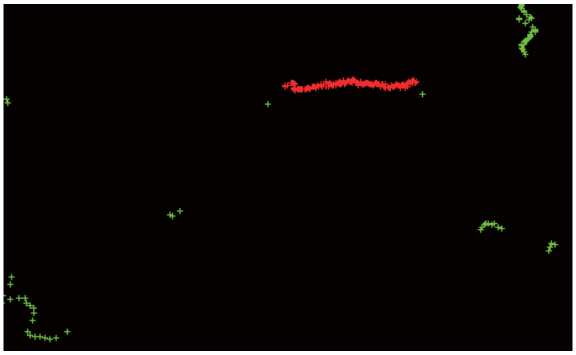


Figure 5. An example of clustering that has identified all laser data that belong to the fence, from among other nearby objects.

the model so that ICP can be effectively applied. A line is fitted to the points of the data cluster. Using this line and the centroid of the cluster, the data points are translated and rotated for a rough fit with the model.

Finally, ICP is performed to match the roughly aligned cluster to the model of the laser fence. Figure 6 shows an example of the data and model alignment before and after ICP is performed. Even though the data is defined to be planar in the x-y plane, ICP occasionally introduces a 180 degree roll about the x-axis because of the symmetry in the line model. This flip can cause problems when used to compute tracking motion commands. Therefore, the current algorithm is enhanced to detect this flip and correct it.

Several additional checks are put in place to reduce the possibility of determining an invalid pose, which could cause the controller to react unpredictably. First, when the number of data points in the cluster is too small (less than 50), a pose is not computed because the clustering algorithm may have selected the wrong cluster of data. Second, the average distance between each point in the cluster and the closest point in the model is computed. This average distance quantifies the fitting error between the model and the data. When the fitting error is greater than a threshold (10mm), a pose is not computed due to a poor fit between the data cluster and the model.

Although this pose estimation algorithm has proven to be successful when the laser measurement data are well behaved, at least two problems were discovered that can cause poor pose results. The first problem is due to misalignment of the laser scanner. The laser fence is approximately 100mm high. When the laser scanner is further than 1.5 meters from the laser fence, the alignment of the laser scanner and levelness of the floor can easily cause the laser beam to miss the fence, and strike the base of the task board instead (see Figure 7, top). A simple solution is to build a taller laser fence so that the laser beam can be ensured to scan the fence area as far as 3 meters away.

The second problem is due to noise and bias in the laser data. At distances closer than about 1m, the range measurements are very noisy and the range values are biased towards the laser. This causes an observable "bowing" in the data (see Figure 7, bottom). This "bowing" effect causes the model fitting to yield a fitting error that is often large enough to fail the final fit check. Currently, we have found no effective solution to this problem, although it is possible to cover the fence with less reflective material to reduce the "bowing" problem, since it helps to reduce the intensity of the laser returns. However, this may not enable the fence to be seen at the required distance of 3 meters.

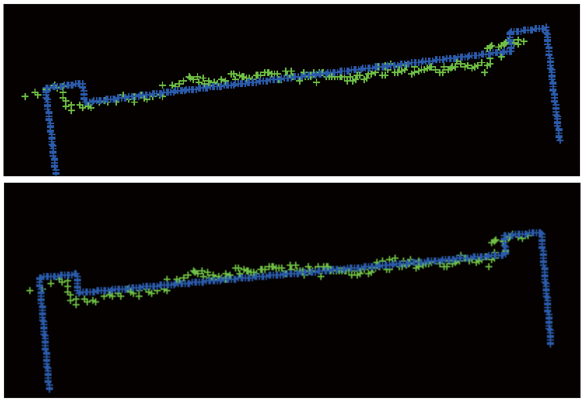


Figure 6. Top: Rough alignment of model before ICP Bottom: Model match after ICP.

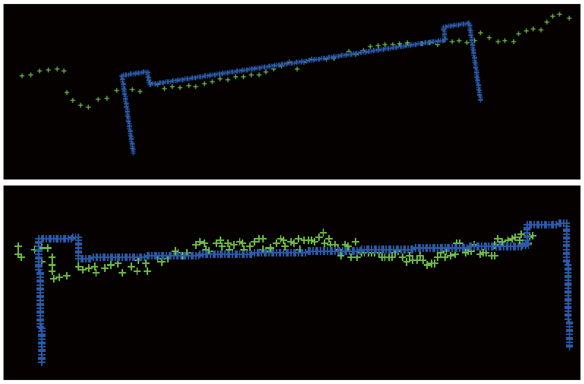


Figure 7. Top: Laser misalignment can cause the laser scan to miss the laser fence and strike the task board itself. Bottom:"Bowed" laser data, due to noise and bias.

IV. Mobile Robotic Tracking

As discussed in the introduction, only the mobile base is used to follow the moving task board. A servo behavior takes as input the pose estimates of the task board relative to the robot, either from laser or visual tracking, and outputs pose commands to the robot, with the goal of moving the robot to a waypoint relative to the task board. We use the pure pursuit algorithm [12] to convert base goal positions into velocity commands. This algorithm controls the base's angular velocity to attract to the goal position and heading.

$$V_t = K_v * D_x \tag{1a}$$

$$\dot{\vartheta} = K_\vartheta * \Delta\vartheta + K_y * V_t * \sin(\Delta\vartheta) * D_y \tag{1b}$$

Where:

V_t is the translational velocity being commanded;

D_x is the translational distance to the goal point;

$\dot{\vartheta}$ is the angular velocity being commanded;

$\Delta\vartheta$ is the difference between the vehicle's angle and the goal angle;

D_y is the perpendicular distance to the goal;

K_x, K_y, and K_θ are proportional gains

Figure 8 illustrates the terms used in the pure pursuit algorithm. The translational velocity in (1a) is proportional to the X distance from the goal point. The angular velocity in (1b) is composed of two terms. The first term is proportional to the heading difference. The second term attracts the vehicle to a line extending from the goal angle: as the vehicle gets closer to the line, the sine term goes to zero, and then the goal angle attraction term dominates.

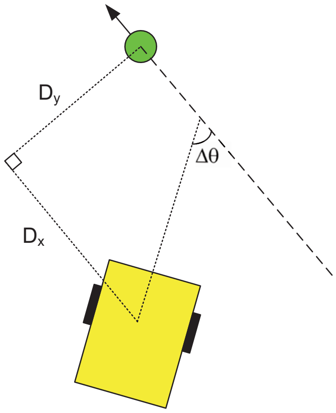


Figure 8. Terms used in the pure pursuit algorithm

To remove the effect of noisy pose estimation, we apply low-pass filters that dampen high-frequency signals by averaging over prior values on the signal. For a new data r_t and a smoothing factor k, the filtered r'_t is defined as:

$$r'_t = (1-k)r'_{t-1} + kr_t \quad (2)$$

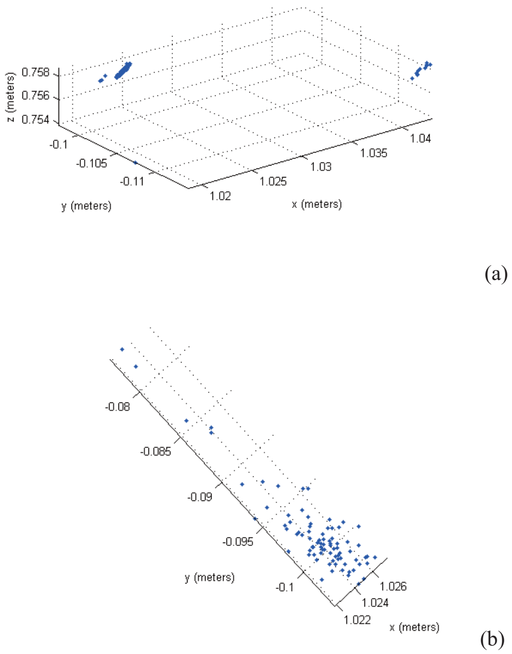
We apply the low pass filter to the outputs of the pose estimation algorithms as well as to the outputs of the pure pursuit algorithm.

V. Experimental Results

A. Static Pose Estimation Results

First, we compare visual and laser pose estimation in the static case. Figures 9 and 10 compare two pose outputs with the robot stationary at one and two meters from the task board. As described in Sections II and III, the visual pose estimation can produce pose outputs at up to 20 Hz, while the laser system outputs at approximately 5 Hz. A total of 30 seconds worth of readings are shown in each case.

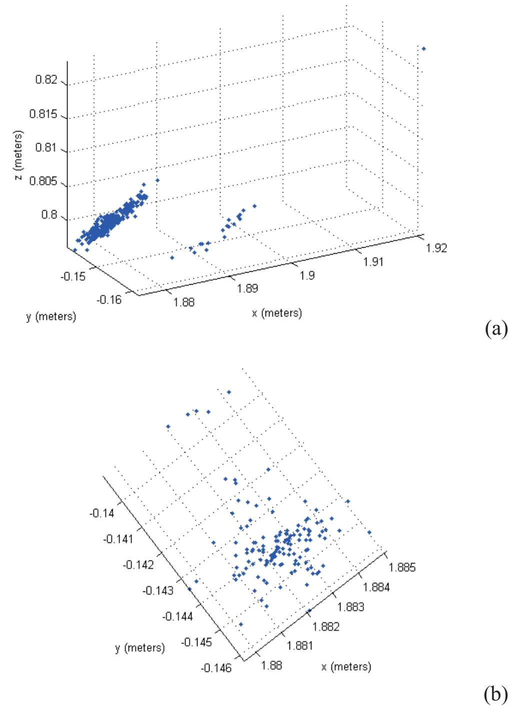
Relative pose estimates using visual fiducials have tighter clusters (one standard deviation is (3, 1, 2) mms in (x, y, z) respectively) compared with the laser scanner (one standard deviation is (1, 4) mms in (x, y) respectively), however multiple distinct clusters exist. In addition, it was discovered that the visual pose outputs jump from one cluster to another only once. The cause and condition for multiple pose clusters are unknown at this point. From these results, we expect the laser tracking system to perform slightly better, in that it will not produce jumps in between cluster readings.



(a)

(b)

Figure 9. Comparison of visual (a) and laser pose estimation (b) with the base fixed 1 meter away from the motionless task board. Each case shows 30 seconds worth of readings. The visual pose estimation system has tighter clusters, but multiple distinct clusters. The laser system precisely locates in the X direction, but varies by more than 2 cm in the Y direction.



(a)

(b)

Figure 10. Comparison of visual (a) and laser pose estimation (b) with the base fixed roughly 2 meters away from the motionless task board. Each case shows 30 seconds worth of readings. The visual system again has multiple clusters. The laser system tracks more precisely than it did at 1 meter, due to increased laser range accuracy from this distance.

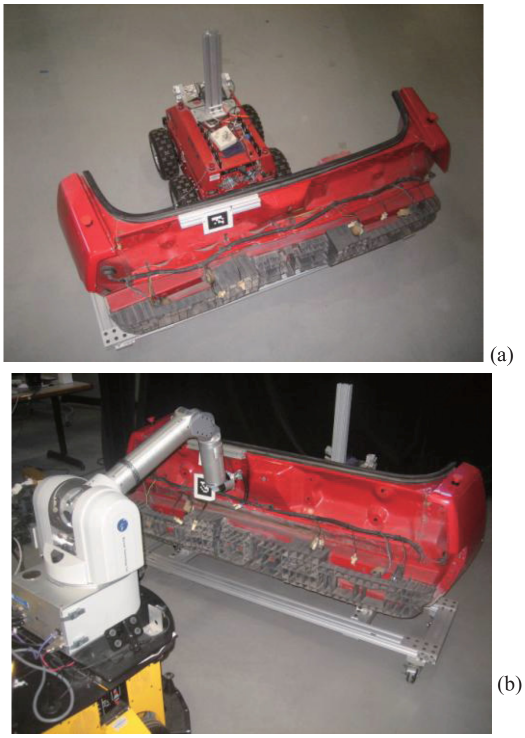


Figure 11. (a) An ATRV-Jr. robot is commanded to move the task board at a constant speed. (b) The WAM arm end-effecter keeps contact while the mobile base is tracking the task board movement using a pure pursuit algorithm with a low pass filter.

B. *Dynamic Tracking Results*

Our task board (see Figure 11a) is the partially-complete back end of an automobile, at a stage when small electrical and mechanical components would be inserted. To simulate the moving assembly line, it is pulled at a constant rate of 3 cm/s by an ATRV-Jr robot. Our mobile manipulator (Figure 11b) is a Barrett WAM arm mounted on a MobileRobots Inc. PowerBot. The base is commanded to track a waypoint 1 meter from the task board using the pure pursuit algorithm. To demonstrate the feasibility of fine-control assembly tasks, the robot arm is simultaneously commanded to maintain constant contact with the task board using a force controller. If the force is too little, the arm controller pushes towards the task board; if the force is too great, it pulls away.

We performed tests of the dynamic tracking system using both the visual and laser pose-estimation systems. Figure 12 compares the linear and angular velocity commands that were sent to the mobile base from the pure pursuit algorithm after low pass filtering. The linear velocity commands were similar in both visual and laser cases. A few larger spikes are observed in the visual tracking. This is consistent with the visual pose data in the static case, as shown in Figures 9(a) and 10(a), in which the visual system shows a large jump in the x-direction, forward from the base. This direction is controlled by speed commands per Equation (1a).

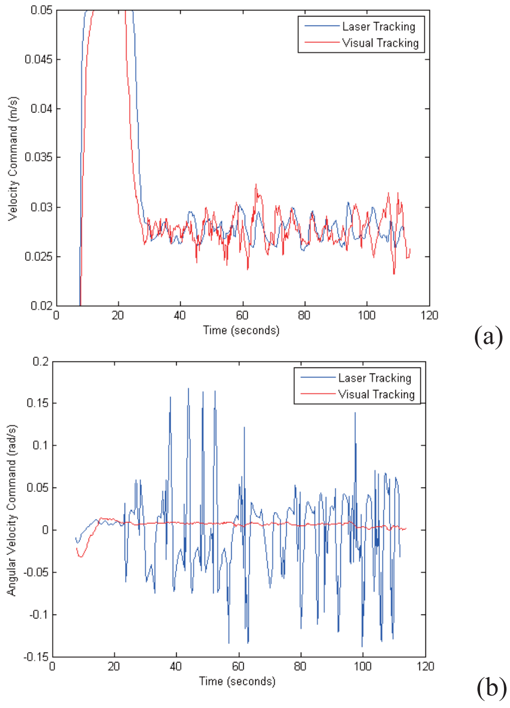


Figure 12. (a) Linear velocity and (b) angular velocity commands sent to the mobile base. Blue commands are using the laser pose estimation system and red commands are using the visual pose estimation system. Both systems send similar commands in translational velocity. The laser tracking system, though, results in highly fluctuating angular velocity commands.

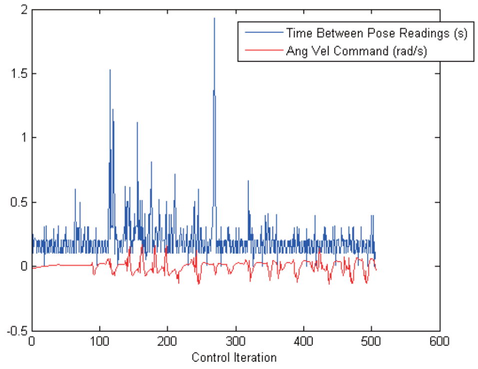


Figure 13. Spikes in angular velocity command are correlated to gaps of up to two seconds between pose estimates from the laser system. During these gaps the base drifts off course, causing a large correction when a new target position is finally received.

The dynamic tracking system proved successful using visual pose estimation. The base control was smooth and the arm was able to keep constant contact with the task board in all tracking tests, as shown in Figure 11(b). The arm's gripper tip slid across the task board surface, but never lost contact with the task board while tracking. This is promising for performing assembly tasks while tracking.

However, base control when using the laser pose estimation was not smooth. The arm's gripper frequently lost contact with the task board. As shown in Figure 12(b), the angular velocity commands vary greatly, exhibiting large spikes. This suggests that D_y and $\Delta\vartheta$ estimation by the laser tracking system varies quite a bit, since a large jump in the left-right direction will result in a large angular velocity command per Equation (1b). We have determined that infrequent pose updates are the source of this problem. The laser scanner is producing raw measurement data at 10 Hz, but the laser pose estimation system produces a pose output only when it can ensure the validity of pose estimate, as described in Section 3. Although the normal rate of laser pose estimation is around 5Hz, there were frequent, large time lapses between subsequent valid pose outputs, as shown in Figure 13. During these lapses the base may drift away from its target pose. When a new valid pose is received, the system must send a large command to the mobile base in order to catch up with the moving task board. The time lapses between valid laser pose estimates could be caused by a combination of factors such as the size of laser fence, imperfections in the floor, as well as pitching of the mobile base during motion.

VI. Conclusions and Future Work

For robots to find use on moving assembly lines, a key challenge is to achieve robust tracking with a reliable, accurate, and fast pose estimation system. We have demonstrated visual and laser pose-estimation systems that meet these criteria. We have developed a model for mobile manipulator control in which a mobile base alone tracks a target using these systems, freeing the manipulator arm for finer control tasks. The experiments to date show promising results that could serve as a foundation for a mobile robot to perform assembly tasks on a moving task board. Future work includes utilizing force control and motion control on a robot arm in these dynamic situations. The ultimate goal is for the mobile robot to perform assembly tasks on a moving task board autonomously, robustly, and reliably.

References

[1] G. N. DeSouza and A. C. Kak, “A Subsumptive, Hierarchical, and Distributed Vision-Based Architecture for Smart Robotics”, *IEEE Transactions on Systems, Man, and Cybernetics -- Part B: Cybernetics*, vol. 34, pp. 1988-2002, October 2004

[2] T. Chang, T. Hong, M. Shneier, G. Holguin, J. Park and R. Eastman, “Dynamic 6DOF Metrology for evaluating a visual servoing algorithm,” Proceedings of the Performance Metrics for Intelligent Systems Workshop (PerMIS), R. Madhavan and E. Messina (eds.), NIST Special Publication 1090, August 2008

[3] J. Shi, “Preliminary Analysis of Conveyor Dynamic Motion for Automotive Applications”, Proceedings of the Performance Metrics for Intelligent Systems Workshop (PerMIS), R. Madhavan and E. Messina (eds.), NIST Special Publication 1090, pages 156-161, August 2008

[4] J. Shi, R. Rourke, D. Groll, P. Tavora, “Quantification of Line Tracking Solutions for Automotive Applications**”,** Proceedings of the Performance Metrics for Intelligent Systems Workshop (PerMIS), R. Madhavan and E. Messina (eds.), NIST Special Publication 1090, pages 189-199, August 2008

[5] B. Hamner, S. Koterba, J. Shi, R. Simmons, S. Singh, “An Autonomous Mobile Manipulator for Assembly Tasks”, accepted to appear in *Autonomous Robots*, 2009.

[6] O. Khatib, “Mobile manipulation: The robotic assistant”, *Robotics and Autonomous System,* vol.26, pp. 157-183, 1999.

[7] J. Tan and N. Xi, “Unified model approach for planning and control of mobile manipulators”, *Proc. of the International Conference on Robotics and Automation*, pp. 3145-3152, 2001.

[8] J. Tan, N. Xi, and Y. Wang, “Integrated Task Planning and Control for Mobile Manipulators”, *International Journal of Robotics Research*, 22(5), pp. 337-354, 2003

[9] Y. Yamamoto and X. Yun, “Coordinating locomotion and manipulation of a mobile manipulator“, *IEEE Transaction on Automatic Control* , 39(6), pp. 1326-1332, 1994

[10] B. Bayle, J.-Y. Fourquet and M. Renaud, “Manipulability of wheeled-mobile manipulators: application to motion generation“, *International Journal of Robotics Research*, 22(7-8), pp. 565-581, 2003.

[11] D. Shin, B. Hamner, S. Singh and M. Hwangbo, “Motion planning for a mobile manipulator with imprecise locomotion“, *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.847-853, 2003

[12] R. C. Coulter, “Implementation of the Pure Pursuit Path Tracking Algorithm”, Technical Report, CMU-RI-TR-92-01, Robotics Institute, Carnegie Mellon University, January, 1992.

[13] P. J. Besl and N. D. McKay, “A Method for Registration of 3-D Shapes”, *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 14(2) pp. 239-256, February 1992.

[14] 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*, 24(6) pp. 381-395, June 1981.

[15] [ARTTag] http://www.artag.net

[16] [Bumblebee 2] http://www.ptgrey.com/products/stereo.asp

[17] [Sick Laser S300] http://www.sick.com/gus/products/new/s300/en.html.html