Obstacle Detection and Avoidance from an Automated Guided Vehicle

Roger Bostelman, Will Shackleford, Geraldine Cheok

***Abstract*— Current automated guided vehicle (AGV) technology typically provides material handling flow along single or dual opposing-flow lanes in manufacturing and distribution facilities. An AGV stops for most any obstacle that may be in its path which then halts other AGVs behind it until the obstacle is removed. An alternative to serial AGV flow is to provide parallel flow in particular areas, such as buffer zones and appropriate lanes where a stopped AGV can be passed by other AGVs. This paper describes two obstacle detection and avoidance (ODA) methods developed and tested. These methods will allow current off-the-shelf AGVs to advance towards unstructured environment navigation.**

I. Introduction

Automated guided vehicle (AGV) technology has been used since 1953 [1] for material handling in manufacturing and distribution facilities. Tug-, unit-load-, and forklift-style AGVs are readily available with typical onboard low-level control of drive, steer, position sensing, guidance sensing, obstacle detection, emergency stop and automatic restart, and start/stop controls among other capabilities. Today, occurrence handling, for example when an obstacle is detected in the vehicle path, is mostly handled locally by the onboard safety sensors – two dimensional (2D), laser detection and ranging (LADAR) sensors – that directly control the vehicle to slow and/or stop via direct electrical connection to the drive amplifiers. Non-contact and/or contact (bumpers) safety sensors are mandatory onboard AGVs, according to the American National Standards Institute/Industrial Truck Standards Development Foundation (ANSI/ITSDF) B56.5 [2] AGV safety standard, where sensors must provide low-level stop capability prior to the AGV structure contacting an obstacle.

Typically, centralized [3] off-board higher-level controllers command AGVs through wireless communication that provides waypoint positions, segment information between waypoints, navigation method and handling, traffic management (e.g., admittance into or decline movement into a particular facility zone), etc. Many AGVs navigate by triangulating laser-based detection of reflectors mounted on walls, resulting in centimeter or smaller repeatability. The AGV movement along segments is programmed into the controller with knowledge of speed, steer method (e.g., Ackerman or quad-steer), onboard equipment adjustment, etc., from one waypoint to the next. Segment information is sent to the onboard AGV controller, typically as the AGV approaches upcoming segments. This ensures that the AGV does not have the entire facility navigation plan that may be uninterruptable and that forces the AGV to follow without updates.

AGVs transporting material usually travel along single or dual, opposing-direction lanes. Therefore, when an AGV halts for an obstacle in its path, it serially stops the flow of other AGVs behind it until the obstacle is removed on its own (e.g., a person walking) or by an AGV supervisor (e.g., a piece of broken pallet dropped in the lane). Workers can anticipate AGV flow in known directions and lanes and at known rates. This method of material handling flow provides intuitive movement for nearby workers and operations. However, this method potentially slows production rates and may require more or faster vehicles to achieve the continuous material flow rates desired by the facility owner. Obstacle detection and avoidance using mobile robot systems is well known in the literature, as a simple internet search illustrates. However, this is not so for AGVs, where only two instances in our search provided examples. One AGV company stated "autonomous navigation provides increased responsiveness, operational flexibility, and improved material flow." [4] Another company demonstrated a floor cleaning robot that navigates around an obstacle in an open area. [5]

The National Institute of Standards and Technology's (NIST) Smart Manufacturing Program has been researching AGV control for developing safety and performance test methods for several years [6, 7, 8]. NIST mobile autonomous vehicle research has investigated performance of obstacle detection algorithms and sensors with respect to standard test pieces, human forms, and overhanging obstacles to foster more intelligently controlled AGVs. Past AGV controls research was enabled through open-source controls and algorithms developed by NIST.

Recently, NIST procured an industrial AGV with stock controls for developing performance metrics and test methods for mobile robots within smart manufacturing facilities. These newer manufacturing settings may have minimal infrastructure, with humans working in close proximity to robots, and may require AGVs to carry advanced onboard equipment such as robotic arms. Using existing technology to conduct this research dramatically reduces the risk to current AGV users and manufacturers. The 2025 Material Handling and Logistics Roadmap suggests that "as confidence in algorithms increases, many routine and even complex decisions will be turned over and automated" and "real-time optimization algorithms for dynamic control of logistics systems should be developed and widely used." [9] Detecting and avoiding obstacles may be considered complex for some in the industry, although it directs them towards future unstructured environment navigation even with their current systems.

Roger Bostelman, Will Shackleford, and Geraldine Cheok are with the National Institute of Standards and Technology, Gaithersburg, MD 20899 USA (phone: 301-975-3426; fax: 301-990-9688; e-mail: roger.bostelman, will.shackleford, geraldine.cheok@nist.gov).

Currently, AGVs typically have closed-source proprietary controls. NIST is investigating whether current, commercial off-the-shelf (COTS) AGVs could be controlled to perform with greater flexibility, such as adapting to changing environments, and if so, how well they perform. Should the AGV adapt appropriately, overall factory performance would benefit, for example, through enabling parallel material handling AGV flow, where a stopped AGV or obstacle could be passed by other AGVs in buffer zones or adjacent lanes. Providing a centralized traffic manager with knowledge of AGV intent to pass obstacles or other AGVs is an obvious issue that was only briefly addressed in recent NIST research and left for future research.

This paper describes two obstacle detection and avoidance methods developed and tested at NIST using an industrial AGV and controller. One method used alternative pre-planned paths drawn on a layout tool offline. The other method planned paths around obstacles after the positions of obstacles were detected during run-time. Multiple AGV control is briefly discussed, followed by performance measurements compared to ground truth.

II. AGV Obstacle detection and Avoidance Algorithms

A. Using Predetermined Paths

An obstacle detection and avoidance algorithm, called obst_avoid, was written and implemented using an NDC8 transport structure [10]. This algorithm was executed when the onboard safety sensor slow field (detection area causing the AGV to slow speed) detected an obstacle in its main segment or path (in this case straight ahead). Paths are defined in a layout application designer, similar to a computer aided design (CAD) software system, that defines the AGV paths and stop points. Upon obstacle detection, the modified controller uses this information to redirect the AGV to a new, predetermined path positioned to drive around the obstacle. The obstacle avoidance path can be an adjacent AGV lane or a buffer area used for this option. The following strawman control algorithm includes obstacle detection, avoidance, and high level AGV controller alert when more than one AGV is being used in the same facility. The following outline describes how the algorithm works when the AGV detects an obstacle in its path, where the waypoint numbers used in the algorithm are shown in Figure 1 (a). Figure 1 (a) also shows the AGV path, points, and obstacle.

1. Drive along typical path from point 1 to 6 - predetermined points.
2. Safety sensor detects an obstacle (red dot marks detect location) in the path within the slow field detection area when the AGV is between point 1 and point 2. Slow the AGV.
3. Send obstacle detect alert to the high level AGV control system (HL-AGVS) (if there are other vehicles in the same facility/zone) of an obstacle in the path.
4. Vehicle control receives a reply from HL-AGVS, of no approaching AGVs – clear adjacent lane or obstacle-passing area, to move from segment 1-2 to point 6 via point 4.
5. Future option: Continuously monitor with onboard safety sensors that the lane/area is clear. Method: Use raw sensor data to determine if obstacles are in the path and to plan well in advance of the AGV slow field detection area.
6. Left/right crab or steer into approaching lane or clear area, from segment 1-2 to point 5 via segment 3-4.
7. Drive past obstacle. Future option: detect using side and rear sensing that the obstacle has been passed and that the AGV lane is clear to reenter.
8. Right/left steer/crab into original AGV lane towards point 6 via segment 4-5.
9. Continue on to previously commanded goal (point 6)
10. End.

Note: If the obstacle was detected when the AGV was between points 2 and 3, the vehicle would still move along segment 3-4 because the AGV has not reached the next segment choice (3-5 or 3-4) However, if the obstacle was detected after point 3 or beyond, the vehicle would stop using the safety sensor stop field detection and control.

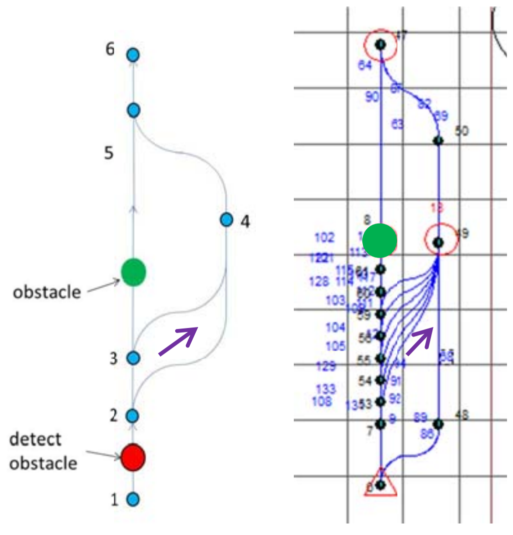


Figure 1 – (a) AGV path, points and obstacle for strawman algorithm development. (b) Layout application plan showing the predetermined AGV paths (blue). The straight vertical path is the commanded path. The red triangle is the start location; the green spot is the obstacle; violet arrows show the obstacle avoidance path; the grid size is 1 m.

The experimental results from several trials showed that the obst_avoid control using a layout application (see Figure 1 (b)) and algorithm performed well: the AGV was able to avoid an obstacle detected using its safety slow field sensor by following a newly-commanded path to drive around the obstacle. Figure 1 (b) shows the paths directly through the obstacle and around the obstacle. No emergency stop was required or automatically occurred and the AGV reached the point 6.

The algorithm did not cause the vehicle to stop at the beginning of each obstacle avoidance start point, therefore traffic flow continued. However, when no obstacle was in the path, the AGV hesitated at each start point. This is because the vehicle was programmed to stop at each point, wait for data on whether there was an obstacle ahead, time for data to reach the traffic manager, and then time for the traffic manager to respond with the 'go-ahead' to the next point. The vehicle could have been programmed to go through multiple points without stopping, but then obstacles would not have been avoided. It is possible that this does not occur with all AGVs, although if so, this control method is disruptive to AGV movement. Other problems with this method include:

1. Only two alternative paths are considered. It may be that both paths are blocked by obstacles and a third path might allow the AGV to successfully avoid the obstacle.

2. Communications delays and the limited range at which an AGV could reliably detect an obstacle limit the speed at which the AGV could be traveling when it detects the obstacle in order to switch to the alternative path before reaching the point that the two paths diverge.

3. More work is required of the AGV installer to calculate and verify the large number of possible alternative paths ahead of time. A smaller number of paths could be calculated, but that would reduce the number of places an obstacle could be detected and successfully avoided.

As such, a different type of obstacle detection and avoidance algorithm was researched using B-Splines detailed in the following section.

B. Using B-Spline Paths

The vendor of the controller for our AGV sells an option to allow the AGV to accept external paths during run-time at the start of a segment. Paths are drawn off-line using the same tool as shown in Figure 1 except some segments are marked as "external". The entire layout, which could include both normal and external segments, is downloaded to the vehicle. A Transmission Control Protocol (TCP) server on the vehicle allows third-party applications, such as the one developed at NIST for our experiments, to monitor the state of the vehicle over a wireless network to know when the vehicle is approaching a segment marked as external. The third party application can then send a list of up to 20 control points (X, Y locations) over that same TCP connection to be used to generate a B-spline that the vehicle will then follow.

There are two significant restrictions:

1. As with most commanded segments, the spline must also start at the original layout start position for that segment and end at the original layout end position. The algorithm can replan up until the start of the blocked segment as the facility sensors provide continuous obstacle position information. However, if the end point of that segment is blocked by an obstacle but some further point in the layout would have been reachable, no replanning is performed since the end-point cannot be bypassed. In other words, the algorithm cannot replan to the final goal because the end point is blocked by an obstacle, nor can it replan to some segment-start closer to the goal.

2. The spline cannot be changed after the vehicle begins following it. This means that new sensor data on the size, shape, and/or position of the obstacles obtained as the spline is being or has already been executed cannot be used to redirect the vehicle around the obstacles. However, should the obstacle move into the new spline AGV path, the onboard AGV safety sensor will still function normally and slow/stop the vehicle prior to contact with the obstacle.

The NIST Spline Generator consists of two parts: obstacle detection and path planning. Obstacle detection reads raw data from two external (i.e., not on the AGV) laser line scanners. A single laser line scanner can provide an estimate of the position of the front of obstacles but since the obstacles are only seen from one side, the size and shape of the back side would be unknown. Therefore, we used two line scanners to measure the size and shape of the obstacle. Figure 2 shows photos of the facility layout, AGV, obstacle, and one of the line scanners. More line scanners could be used to better cover a larger area. The scanners were mounted, with variable height, to posts set on the floor and used to scan in a horizontal plane a few inches above the floor. Since the scanners detect obstacles only in a single plane a few inches from the floor, they can detect things such as the legs of a table or ladder. Other line scanners could be mounted on a motorized tilt or a 3D sensor could be used to detect higher or overhanging obstacles.

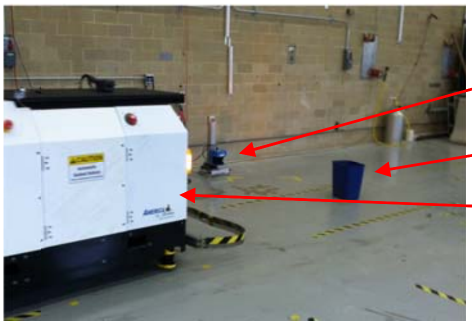


Figure 2 shows photos of the facility layout, AGV, obstacle and line scanners.

Using external sensors rather than sensors mounted on the AGV has several advantages because external sensors:

1. Can be used to provide information for multiple AGVs.
2. Can be placed so that obstacles can be detected from more than one direction.
3. Can be changed without modifying the AGV, adding any additional weight, drawing power, and/or changing the footprint of the AGV (i.e., sensor on an AGV could protrude from the vehicle causing a potential hazard).
4. Uncertainty in the position of the vehicle does not add to uncertainty in the position of the obstacles. Unfortunately, although the vehicle would seem to have an accurate position estimate of the AGV for use internally, this does not appear to be available for third party applications through the TCP connection previously discussed. Even if it were available, unexpected delays in receiving the position could result in errors in the position associated with the obstacle.

The primary disadvantage of mounting the line scanners throughout the facility is that if there is a large area to cover and only a few AGVs, then mounting the sensors to the room will require more sensors than using a sensor mounted to the vehicle. Ideally, safety line scanners that are typically used for detecting and slowing/stopping the vehicle when obstacles are in the vehicle path would also provide relative position information of obstacles beyond the slow and stop zones. Today, there are such sensors available in the market. To use this feature would require simultaneous AGV position and relative obstacle position to the AGV from the onboard line scanners and a filter to only detect obstacles in the AGV path.

Figure 3 shows a graphical display of the obstacle detection program. The obstacle detection filters out background points, through use of a simpler set of fixed boundaries, corresponding to the walls and fixed equipment that the motion planner will avoid. Filtering out these points reduces the computational work-load for the planner.

The range points from the line scanners are converted to 2D points in the room using the calibrated position of the sensor and then grouped so that the planner only needs to deal with a rather small set of obstacles. It also filters out dynamic obstacles or obstacles that have not remained at the same position for the last five seconds. It generally makes more sense to wait for dynamic obstacles, such as people or other vehicles, to move out of the way than to plan around them. It is important to understand that the AGV still has the original safety system running unaffected by the obstacle avoidance system. That safety system, and not the obstacle detection system, is responsible for stopping the vehicle to prevent injury to people or equipment.

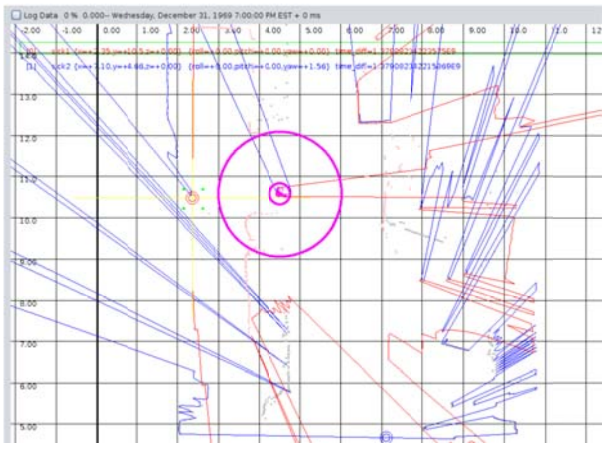


Figure 3: Graphical Display of Obstacle Detection program: red lines show range data from one line scanner, blue lines show range data from the other line scanner, the detected obstacle is shown with two concentric purple circles

Path planning takes the following as input:

1. List of obstacles detected from the current sensor data. Each obstacle is associated with an X, Y center point and a radius.
2. List of boundaries drawn off-line by hand using the room's CAD drawing. Each boundary is associated with two X, Y locations for the two end points of the line segment.
3. The start position and orientation for the next external segment.
4. The end position and orientation for the next external segment.
5. Whether the segment will be performed in crab mode or not.
6. The dimensions of the vehicle's footprint. This includes the distance from the commanded point to the farthest point in front of the vehicle that an obstacle could trigger an emergency stop, as well as the distances to the farthest points to the rear of the vehicle and to each side.

The planning algorithm is based on the common A* search where the nodes in the graph represent candidate X,Y control point locations and the edges are weight costs based on the distance between the points. Edges are only added if the spline between the two points could be traveled without any part of the vehicle's foot print entering an obstacle radius or crossing a boundary.

The algorithm uses two lists of points. One list contains unopened planner points whose reachability from the start location is unknown. The other list contains opened points that are reachable from the start location and have at least one untested potential next point. The unopened list initially contains the goal point and points strategically placed around each obstacle or boundary. The opened list initially contains only the start position. Each point also has an associated list of untested next points. For the start point, this is initialized with a copy of the unopened points list.

A loop is repeated until either a path to the goal is found (see Figure 4) or there are no more points on the opened list. Within the loop, the best point on the opened list is determined as the point with the lowest value of the heuristic function. The heuristic function is the distance of the shortest obstacle-free path found so far to that point plus the minimum over all untested next points of the distance to that point plus the distance to the goal from that point. The path from the best point from the open list to the best untested next point from the list for that point is checked. The check determines if an obstacle or boundary would be encountered by the area swept out by the AGV while moving between the two points.

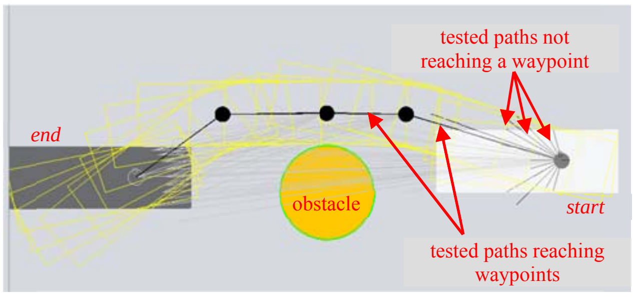

tested paths not reaching a waypoint
end
obstacle
start
tested paths reaching waypoints

Figure 4: Plot showing the planned AGV path (black line) and tested paths (gray lines to no point) around an obstacle. The white and dark gray filled boxes are the AGV at the start and end points and the yellow boxes are the sweep of the AGV as it moves along the planned path.

If no obstacles or boundaries would be encountered, the point last tested is added to the list of opened points and the points on the unopened list copied to the list of untested next points for the point being added to the opened list. In addition, if the test succeeds and the last point was the goal, the loop can be ended with success. Regardless of the result of the check, the point is removed from the untested next points for the first point taken from the open list and this will result in a change in its associated heuristic value. If the untested next points are now empty, the first point is removed from the opened list and if that makes the opened list itself empty, the loop is ended in failure.

Psuedo code:

```
unopened_list ← [ goal + points around each obstacle and boundary ]
start_point.untested_next_points ← copy(unopened_list)
opened_list ← [ start point ]
while opened_list is not empty :
  best_opened ← find_best_point(opened_list)
  best_next ← find_best_point(best_opened.untested_next_points)
  if test(best_opened,best_next) is ok :
    best_next.prev ← best_opened
    best_next.untested_next_points ← copy(unopened_list)
    opened_list ← [ opened_list + best_next ]
    if best_next is goal :
      end the loop , we found the path to the goal
  best_opened.untested_next_points.remove(best_next)
  unopened_list.remove(best_next)
  if best_opened.untested_next_points is empty :
    opened_list.remove(best_opened)
    if opened_list is empty :
      end the loop, failure no path to the goal
```

The test involves generating a series of polygons for a move and then testing each boundary and obstacle to determine if a boundary has either end point in the polygon or intersects any edge of the polygon or an obstacle is within any polygon or closer to any edge than its radius. Figure 6 shows a screen snapshot of the resulting algorithm planning a path around obstacles in the AGV path detected by facility line scanners.

The algorithm makes several assumptions that may cause the path found to be less than optimal or to fail to find a path even when one was available.

1. It only considers the finite list of points added to the initial unopened point list.
2. It will not check whether a point could be reached if the starting orientation were different. There may be more than one way to reach the same point and therefore more than one orientation the vehicle would have at that point, however the tests to next points will assume the orientation that would result from the first path found to that point.

The path shown in Figure 5 was computed in 54 ms (on an 8 core x 2.6 Ghz laptop). Figure 6 shows a series of snapshots of the AGV performing obstacle avoidance using the spline method.

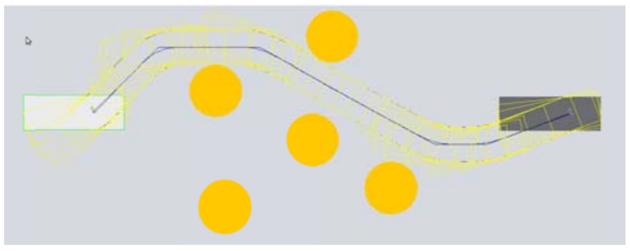


Figure 5: Graphical output of path planner, starting footprint of the AGV is in white, the goal position is dark grey rectangle. Yellow rectangles show the area swept out as the AGV would travel, blue curve shows the resulting spline, orange circles represent obstacles

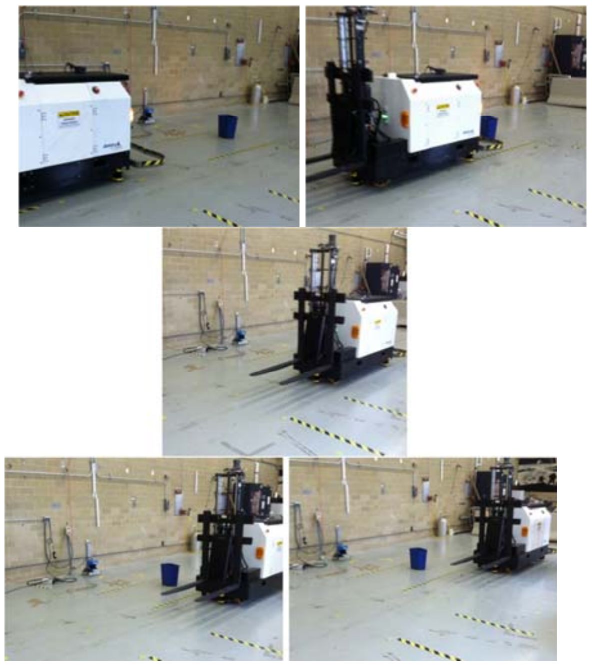
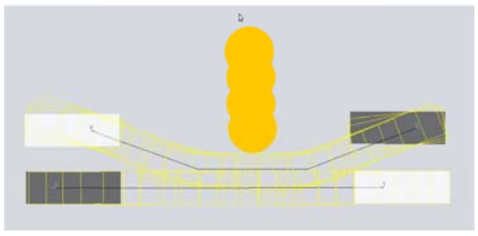
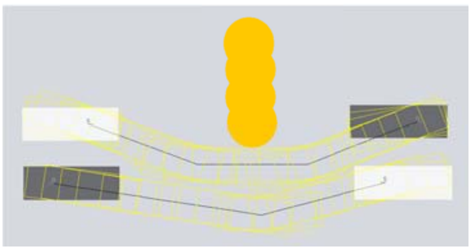


Figure 6: (top-left to bottom-right) Series of snapshots showing the AGV performing obstacle detection and avoidance using the spline method.

III. Multiple Vehicles

Although we only have one test AGV, the effect of one AGV's obstacle avoidance on other AGV is known to be an important issue. A feature called “Exclusive Tracks” was added to prevent AGVs from approaching each other from different directions that likely result in both AGVs sensing the other and going into E-Stop mode. With this option, one AGV is given priority. The path that it sweeps out adds boundaries that will be considered in planning the path of the other AGV. Figure 7 shows the effect of both with and without the “Exclusive Tracks” feature enabled.


a



b

Figure 7: (a) One AGV plan to the right over another AGVs plan to the left shows conflicting areas of the path since the "Exclusive Tracks" option was not selected. (b) The same setup as in (a) with the "Exclusive Tracks" option enabled causing the AGV moving to the left to avoid the conflicting area and reduce the chances of an E-STOP.

IV. Algorithm Performance Compared to Ground Truth

Upon development of the algorithm, the performance of the algorithm was evaluated in an experiment where ground truth measurements were obtained from a laser tracker. It was intended here to mainly compare the best fit path to ground truth, not the interpolated b-spline curve at turns. The laser tracker had an uncertainty of 18 µm at 12 m to track AGV motion (see Figure 8). An SMR (spherical mounted retroreflector) was placed on top of the AGV above the AGV center. The AGV center was defined as the centroid of the four AGV wheels. The planar motions of the AGV, with an obstacle (small or large) in its path, were compared to ground truth measurements.

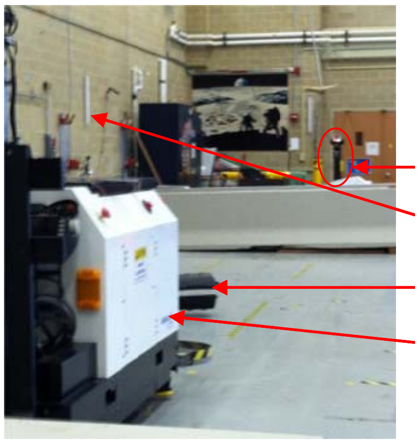
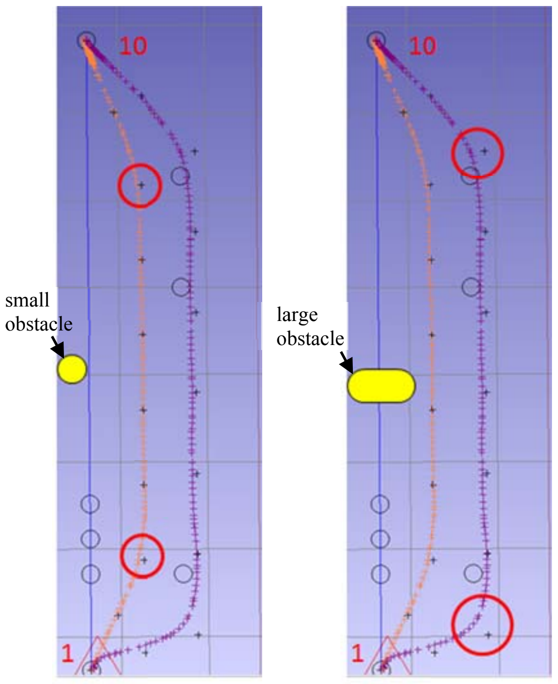


Figure 8: Photo of ground truth measurement system (laser tracker) setup.

A set of six runs were performed with a small obstacle and a set of six runs were performed with a large obstacle. For the set of six runs, three runs were in the 'forward' direction and three runs were in the 'reverse' direction. That is, one run was from waypoint A to B (forward direction) and the next run was from waypoint B to A (reverse direction). Figure 9a shows a plot of the ground truth measurements for one of the six runs with the small obstacle (orange markers), and Figure 9b shows a plot with the large obstacle (purple markers). As seen in the figure, the largest deviations of the AGV from the commanded waypoints occur at the turns (indicated by the red circles in Figure 9).

Figure 10 shows the comparison of the average deviations of the six runs for the small and large obstacle. The error bars indicated one standard deviation of the average. As seen in Figure 10, the magnitude and variability of the deviations are greater for the large obstacle. Excluding the turns, the average deviation for the small obstacle is less than 10 mm and 40 mm for the large obstacle. The maximum average deviations on the turns are 65 mm and 170 mm for the small and large obstacle, respectively.

The commanded points are control points for the b-spline. Control points are typically the exact stop or cross points for AGVs using current AGV path planners. However, control points for a b-spline determine the shape of the curve but are not necessarily expected to lie on the curve. These deviations indicate neither a flaw in the algorithm that generated the control-points nor in the lower level software/hardware responsible for following the curve. This b-spline effect is not related to obstacle avoidance, but only to the tightness of the curve and the inherent characteristics of b-splines. The performance measurement method proved useful to understand the effect and the distance from the b-spline waypoints to the actual path followed by the AGV (i.e., the interpolated b-spline). Exact (interpolated) points to be crossed by the AGV could be calculated when using a b-spline where the performance measurement demonstrated may prove useful and results better fit the interpolated path, especially for AGV navigation in highly confined, unstructured areas.



a b

Figure 9: Measurement results as compared to ground truth for obstacle detection and avoidance using a spline algorithm for runs with a (a) small obstacle and (b) large obstacle. Black '+' symbols are commanded waypoints. Orange and purple '+' symbols indicate ground truth AGV positions for the small and large obstacle runs, respectively.

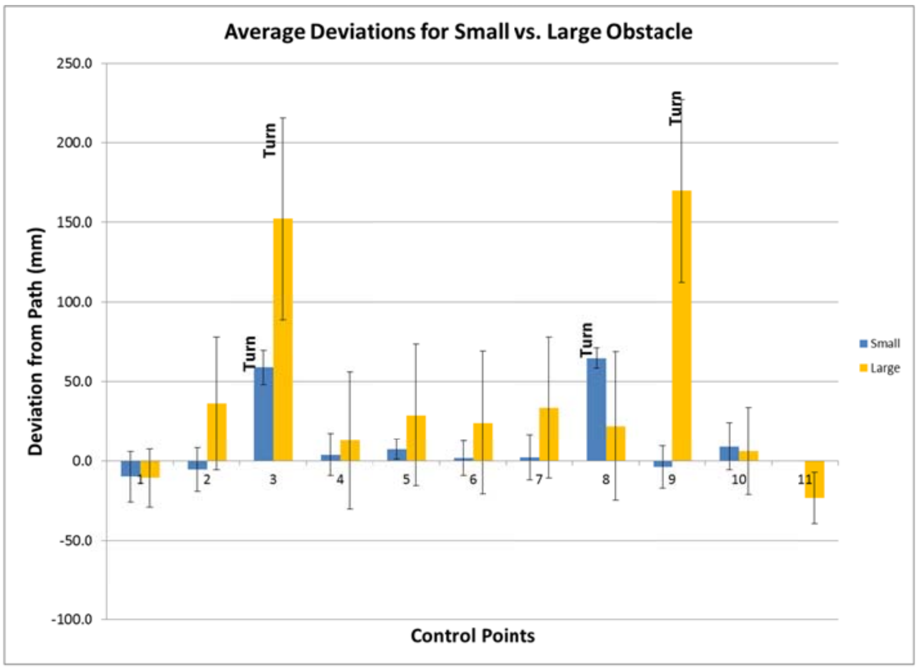


Figure 10. Comparison of the average AGV deviations from the commanded waypoints for a small obstacle vs. a large obstacle.

V. Conclusion

Based on internet searches and conversations with AGV industry representatives, at least two companies have performed obstacle detection and avoidance tests using AGVs. The research also found that other AGV manufacturers described traffic management issues as a low priority issue, although they expected that obstacle detection and avoidance was feasible using current sensor technology. NIST concludes from past research with mobile robots and the 2025 Material Handling and Logistics Roadmap that future benefits to the industry for performing obstacle detection and avoidance would allow: increased productivity for users, perhaps require fewer yet faster AGVs, and facilitate more capable AGVs with minimal modifications to current controllers that would allow navigation through less structured environments.

Two obstacle avoidance control algorithms were designed and tested on the NIST AGV. One includes using many pre-computed paths for the AGV to select from to avoid an obstacle depending upon when the obstacle was detected. The second algorithm uses a b-spline function and higher-level program with facility sensor information about the obstacle position. The second algorithm provides much more control flexibility to avoid more than one obstacle while planning complex paths. Both algorithms could allow adjacent traffic information to minimize risk of AGV to AGV collisions or bottlenecks in AGV traffic. Performance measurement of the B-spline controlled AGV indicated that the largest deviations of the AGV from the commanded waypoints occurred at the turns. Waypoint access tolerance at the turns could, however, be set at a minimum for improved navigation performance between narrow facility passages.

Acknowledgment

The authors would like to thank the AGV manufacturers and users for their input to this research, especially Tommy Hessler from America In Motion.

References

[1] Savant Automation, http://www.agvsystems.com/history-agvs, 2012.

[2] American National Standards Institute/Industrial Truck Standards Develop Foundation (ANSI/ITSDF) B56.5:2012 Safety Standard for Driverless, Automatic Guided Industrial Vehicles and Automated Functions of Manned Industrial Vehicles, 2012.

[3] Y. Uny Cao, Alex S. Fukunaga, Andrew B. Kahng, “Cooperative Mobile Robotics: Antecedents and Directions,” Autonomous Robots 4, 7–27 (1997).

[4] Video of obstacle avoidance, http://youtube.com/JVigBUcvA-c, consulted October 2013.

[5] Video of autonomous floor cleaning robot avoids obstacles, http://www.youtube.com/watch?v=4GJ00EBbBfQ, consulted October 2013.

[6] Roger Bostelman, Will Shackleford, Geraldine Cheok, Richard Norcross, “Standard Test Procedures and Metrics Development for Automated Guided Vehicle Safety Standards,” Performance Metrics for Intelligent Systems (PerMIS’12) Workshop, College Park, MD, March 2012.

[7] Roger Bostelman, Will Shackleford, Geraldine Cheok, Kamel Saidi, “Safe Control of Manufacturing Vehicles Research Towards Standard Test Methods, International Material Handling Research Colloquium (IMHRC) 2012, Gardanne, France, June 2012.

[8] Roger Bostelman, Richard Norcrossa, Joe Falco, Jeremy Marvel, “Development of Standard Test Methods for Unmanned and Manned Industrial Vehicles Used Near Humans, SPIE Defense, Security, and Sensing 2013, Baltimore, MD, May 2013.

[9] Material Handling and Logistics U.S. Roadmap, http://www.mhlroadmap.org/,2014.

[10] NDC8 controller, www.ndc8.com.