Introducing Mobile Robots to Moving-Floor Assembly Lines

Design, Evaluation and Deployment

Vaibhav V. Unhelkar*, Stefan Dörr†, Alexander Bubeck†, Przemyslaw A. Lasota*, Jorge Perez*, Ho Chit Siu*, James C. Boerkoel Jr.‡, Quirin Tyroller§, Johannes Bix§, Stefan Bartscher§ and Julie A. Shah*

ROBOTS that operate alongside or cooperatively with humans are envisioned as the next generation of robotics. Toward this vision, we present the first mobile robot system designed for and capable of operating on the moving floors of automotive final assembly lines (AFALs). AFALs represent a distinct challenge for mobile robots in the form of dynamic surfaces: the conveyor belts that transport cars throughout the factory during final assembly. In this work, we identify the key behaviors necessary for autonomous navigation along dynamic surfaces, develop a control strategy to achieve trajectory tracking on such surfaces, and design a sensing module capable of detecting conveyor belt speed and location in a factory setting. These solutions are integrated with localization, path planning and car tracking to achieve autonomous navigation. The system is implemented via Rob@Work 3 (a robotic platform designed for industrial applications) and Robot Operating System (ROS). The integrated system is evaluated on an operational automotive factory floor alongside human workers. A mobile robot capable of working on moving floors (conveyor belts) alongside human associates can provide greater flexibility when designing automotive manufacturing processes, yielding ergonomic benefit for users, improving task performance and efficiency, and opening avenues for novel application of robotics.

Corresponding author: unhelkar@csail.mit.edu
*Massachusetts Institute of Technology, Cambridge, MA.
†Fraunhofer Institute for Manufacturing Engineering and Automation IPA, Stuttgart, Germany.
‡Harvey Mudd College, Claremont, CA.
§BMW Group, Munich, Germany.

Robots in Automotive Manufacturing

In recent decades, large industrial robots have revolutionized manufacturing across multiple sectors. The automotive industry was among the first to introduce robotics into the manufacturing process; today, approximately half of the manufacturing tasks in a typical automotive factory are performed by industrial robots. However, the majority of these robots are caged, stationary, and non-interactive. A robot's environment is rendered highly predictable through operations away from humans, allowing for the reliable and safe execution of pre-planned tasks.

In recent years, the boundaries for robots within the realm of manufacturing have begun to expand [1, 2], introducing them into the final assembly process to work alongside humans. Intelligent assist devices and stationary robots have been used for car assembly tasks [3]. These robots operate in close proximity with humans; they are stationary, however, and are limited in the operations they can perform and the flexibility they allow for manufacturing processes.

Several automotive factories incorporate automated guided vehicles (AGVs) to deliver parts across large-scale factory floors. However, these vehicles work with their mobility limited to specific grids positioned within the environment, and are incapable of entering the area in which value-added work is performed on the assembly line. "Robot Workmate," a collaborative robotic system for AFALs designed by Müller et al. [4], is capable of performing inspection tasks; however, the system's mobility is limited to preinstalled linear rails within the work environment. Freely moving mobile robots have recently been developed for use at automated workstations [5, 6], but these systems require that work is performed on static surfaces. Consequently, such systems are not compatible with AFALs, which typically include dynamic surfaces in the form of the conveyor belts that transport vehicles during assembly.

To the best of our knowledge, there have thus far been no mobile robots capable of working alongside humans on dynamic AFAL surfaces. In this paper, we report on a robotic system that autonomously navigates along moving-floor assembly lines, and is not restricted to preconfigured paths. Our system opens new opportunities for close-proximity human-robot collaboration during final automobile assembly on dynamic AFAL surfaces. In contrast to systems with constrained mobility, a freely moving robot provides a larger operating region and can potentially improve task efficiency. We highlight this performance benefit through a simulation evaluation of human-robot interaction, in which the system predicts human motion and plans in time to navigate amongst humans in the shared workspace. Finally, we discuss directions of future work for achieving seamless human-robot collaboration in production environments.

A preliminary version of this work, in which the system was demonstrated in a laboratory environment, is available at [7]. Here, we present an integrated mobile robotic system for use in automotive factories that includes algorithms for localization, car tracking and path planning, and demonstrate this system in a workspace shared with humans. As part of this demonstration, conducted in October 2014, the robot operated successfully on a live automotive production line for 1 week (5 work days) and performed a prototypical assembly task on approximately 200 cars. To our knowledge, this is the first instance of a mobile robot navigating dynamic surfaces within automotive factories.

Automotive Final Assembly Lines

The automotive final assembly environment (shown in the title figure) is highly dynamic, uncertain, and includes both humans and mobile objects, such as cars on the assembly line, AGVs, and part carts. The motion of humans is unconstrained: they are free to move throughout the factory both on and off the assembly line.

Automotive assembly lines also feature dynamic surfaces: the conveyor belts that transport cars within the factory. The conveyor belts are flush with the surrounding static surfaces. The speed of these belts - and, consequently, that of the cars they transport - is largely uniform. While in motion, the conveyor belts tend to move at a slow and almost constant speed (< 200 mm/s); however, a belt can occasionally stop due to events occurring on the factory floor.

The motion of the belt results in each car typically spending fewer than 3 minutes at a given workstation. Unscheduled stops are highly discouraged, as they can lead to delays and substantial economic costs (e.g., losses greater than $10,000 per min [8]), rendering the tasks to be performed during final assembly highly time-critical. Any agent involved in automobile assembly should be able to successfully and repeatedly complete tasks within short time cycles and on dynamic surfaces, as incomplete work would result in unscheduled stops.

Challenges for Autonomous Navigation

Here, we identify the key behaviors necessary for a mobile robot to achieve autonomous navigation on AFALs. In order to do so, we focus on the differentiating characteristic of AFALs from other manufacturing environments: the presence of dynamic surfaces.

First, the robot must have knowledge of whether it is currently located on a dynamic surface. This requires that the robot maintain a location estimate of the conveyor belt relative to itself.

Second, awareness of the assembly line's speed is required for trajectory tracking, localization, and tracking objects along the line. This information may be required even when the robot is not positioned on the assembly line.

Third, the robot requires trajectory tracking algorithms in order to follow a desired path along dynamic surfaces. The robot can enter the conveyor belt in any arbitrary orientation, and can have any number of its wheels present on the dynamic surface. Trajectory tracking must also be functional on static surfaces, or while positioned partially on and partially off the assembly line in any orientation.

Solutions to achieve these behaviors ideally should require minimal modifications to factory layout and infrastructure and allow for use of off-the-shelf software and hardware. Further, the robot should be able to plan paths in dynamic

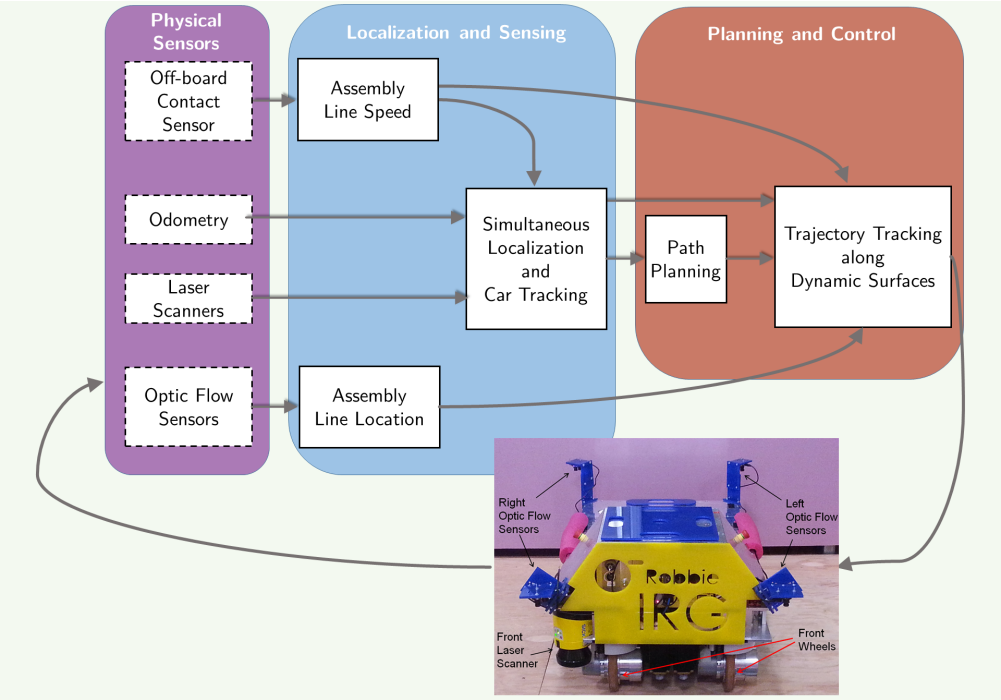


Figure 1: Overview of the mobile robotic system along with an image of the Rob@Work 3 mobile base.

environments and in the presence of human coworkers. Navigation within dynamic environments and among humans is an active area of research [9]. Hence, we provide a modular solution for navigation that not only takes dynamic surfaces into consideration, but also enables the use of existing algorithms for path planning within dynamic environments.

System at a Glance

To achieve the key behaviors highlighted above, we provide (i) an algorithm for trajectory tracking along both static and dynamic surfaces, and (ii) a sensing system for detecting the presence and speed of dynamic surfaces. The successful achievement of these behaviors allows robots operating on dynamic surfaces to make use of localization and path planning algorithms originally designed for use on static surfaces. The developed solutions are integrated with localization and path planning to achieve autonomous navigation during final automobile assembly. We also briefly discuss an approach to tracking dynamic goals (i.e., the cars being assembled on the line). Figure 1 depicts an overview of our mobile robotic system.

We chose Rob@Work 3 (see Fig. 1 and Table 1) as the base robotic platform for our system, primarily due to its four independently actuated wheels that can be both steered and driven. We augmented the Rob@Work 3 platform with additional sensors to detect the speed and location of conveyor belts in an automotive factory. In addition, the Rob@Work 3 platform possesses the following desirable characteristics for facilitating industrial operations:

Table 1: Rob@Work 3: Salient Features

Dimensions	$103 \times 57 \times 40$ cm
Weight	120 kg
Payload Capacity	150 kg
Maximum Speed	1.2 m/s
Actuators	4 wheels (2 motors per wheel, for driving and steering)
Sensors	Eight encoders (1 per motor)
	2 SICK S300 Laser Scanners

- a high payload capacity, necessary for assembly tasks;
- battery life of ≈ 8 hr, enabling extended operation;
- on-board sensing;
- a laser-scanner based safety system; and
- a middleware based on ROS.

Reference Frames and Variables

A schematic of the factory environment (workstation) where we evaluated our system is included in Fig. 2, which also illustrates the reference frames and variables used herein. The dynamic surface (conveyor belt) is depicted in gray. "Start" refers to the initial location of the robot. Points "Goal-A" and "Goal-B" are defined relative to the car currently being assembled.

We use the static coordinate systems of the world W and the conveyor belt B, and the non-static coordinate systems of the robot R and the moving automobiles $I, J, \cdots$. The x coordinate of the transition between static and dynamic

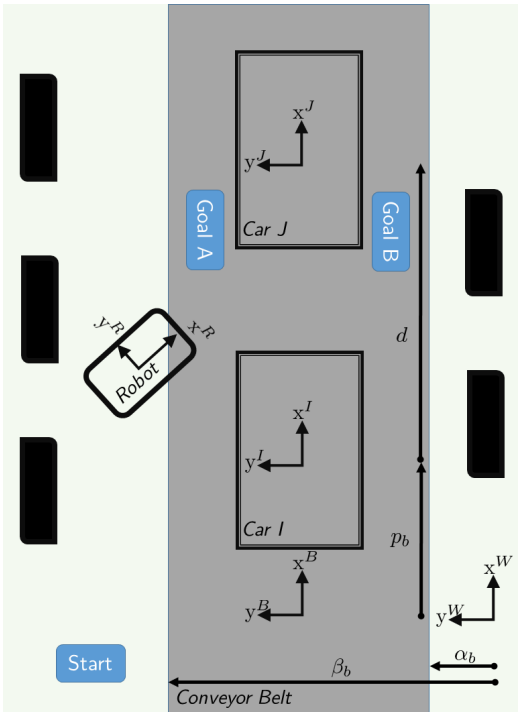


Figure 2: A schematic of the robot's workstation during factory evaluations.

surfaces, as specified in the world frame, is referred to as the location of the assembly line and represented by the variables α_b and β_b. The position of the first car in the belt frame is denoted by the variable p_b, and the constant distance between two consecutive cars is denoted by d.

Trajectory Tracking along Dynamic Surfaces

State-of-the-art wheeled robots are capable of following a given path with high fidelity along static surfaces. In the case of Rob@Work 3 specifically, the nominal control architecture for trajectory tracking incorporates multiple feedback loops [10]. A path planning algorithm or human teleoperator issues a desired trajectory, which is translated into velocity commands for the mobile robot. The desired velocity of the i$^{\text{th}}$ wheel $\mathbf{v}^R_{\text{wheel,i}} = (v_{\text{x,i}}, v_{\text{y,i}})$ is obtained in terms of robot velocity $(\dot{x}_\text{r}, \dot{y}_\text{r}, \dot{\phi}_\text{r})$ as follows:

$$v_{\text{x,i}} = \dot{x}_\text{r} - \dot{\phi}_\text{r} y_{\text{w,i}} \quad (1a)$$

$$v_{\text{y,i}} = \dot{y}_\text{r} + \dot{\phi}_\text{r} x_{\text{w,i}}. \quad (1b)$$

The controller then converts each wheel velocity command to the wheel configuration, a steering angle ψ and an angular rate $\dot{\theta}$ command, as detailed in prior work by Connette et al [10]. A PD controller is used to control ψ and $\dot{\theta}$ for each wheel.

Algorithm 1: A modification to the command issued to the nominal wheel controller for navigating environments with dynamic surfaces.

Input: Nominal wheel controller command, and Surface velocity
Output: Compensated wheel controller command
$\mathbf{v}^R_{\text{surf,i}}$: absolute velocity of the surface at i$^{\text{th}}$ wheel
$\mathbf{v}^R_{\text{wheel,i}}$: absolute velocity of the at i$^{\text{th}}$ wheel
foreach *robot wheel* **do**
  sense absolute surface velocity ($\mathbf{v}^R_{\text{surf,i}}$) at wheel;
  modify the nominal command:-
  $\mathbf{v}^R_{\text{wheel,i}} = \mathbf{v}^R_{\text{wheel,i}}$ - $\mathbf{v}^R_{\text{surf,i}}$;
end

Limitations of Nominal Architecture

The nominal trajectory tracking architecture of mobile bases, however, is not designed to accommodate dynamic surfaces. To illustrate this point, we created a scenario that included a dynamic surface using the Gazebo simulator. The robot's task was to navigate in a straight line across the dynamic surface moving at a rate of 100 mm/s. However, the robot continuously deviated from the desired path, indicating failure of the existing architecture to follow the desired trajectory (a video is available[1] at `http://tiny.cc/mra1`).

Further, by not accounting for the motion of the dynamic surface, the robot experienced torques at the wheels while transitioning from a static to dynamic surface, or vice-versa. Repeated application of these torques would structurally weaken the robot, which would in turn impact system maintainability and be highly undesirable for the effective introduction of mobile robots onto a factory floor.

Architecture for Dynamic Surfaces

We designed a control algorithm based on reference shaping that considers the surface speed as additional input; this allows for modular implementation, but requires additional sensing of surface parameters. However, as this method avoids undesired effects on robot hardware - a key requirement for the structural integrity of robots and their effective introduction onto a factory floor - we adopted this approach.

Fig. 3 depicts our control architecture, and Algorithm 1 details the modification we made to the nominal controller. This architecture leverages the independent actuation of each wheel, and compensates for the motion of the dynamic surface by suitably modifying the reference sent to the robot's wheel controllers. This results in a modular design that preserves the use of the existing wheel PD controllers and soft-

[1]The video is also included as a multimedia attachment with the submission, please see file: VIDEO-CLIP-1.mp4

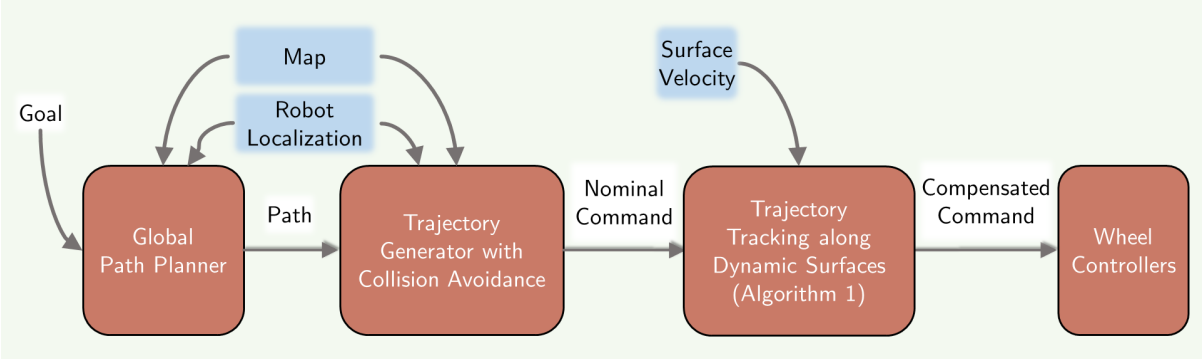


Figure 3: Our control architecture for autonomous robot navigation on dynamic surfaces.

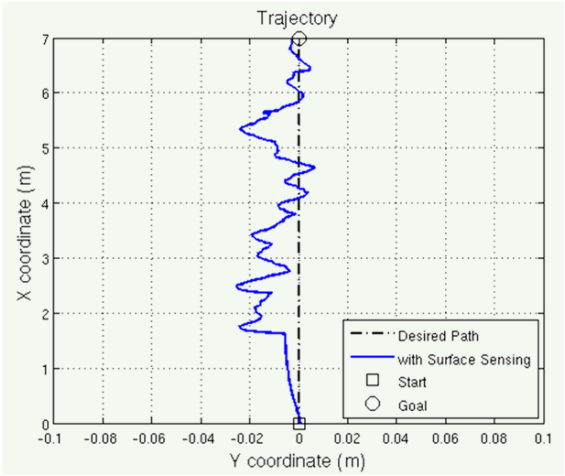


Figure 4: The deviation from the desired path as the robot crosses the moving surface.

ware architecture. Using Algorithm 1, we compensated the command for each wheel ($\mathbf{v}^R_{\text{wheel,i}}$) based on the absolute surface velocity at its point of contact ($\mathbf{v}^R_{\text{surf,i}}$). The modified wheel velocity command is used to compute the wheel configuration - specifically, the steering angle and angular rate.

We validated the designed algorithm using the same Gazebo simulation environment in which the nominal architecture was evaluated. During the task, the robot's deviation from the nominal path remained < 4 cm (see Fig. 4; a video is available[2] at `http://tiny.cc/mra1`). Algorithm 1 enabled the robot to successfully navigate across the simulated assembly line by dynamically compensating for surface velocity and correcting the robot heading accordingly.

Sensing the Assembly Line

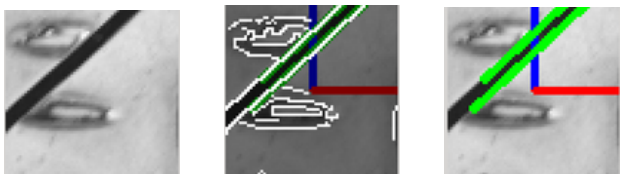
When designing a module for sensing the assembly line, we explored the use of four types of sensors: miniature radars, optic flow sensors, contact-based encoders, and inertial sensors. As the surface in question moves relatively slowly ($<$ 200 mm/s), the performance of miniature radars and low-cost inertial sensors is limited by poor accuracy at low speeds. Further, measurements obtained through an indirect method (such as an inertial sensor-based system) would be reactive, detecting surface motion only through disturbances to the robot's motion caused by the surface.

On-board optic flow sensors have been previously used to maintain location estimates for mobile robots [11] on static surfaces. Also, images from on-board optic flow sensors can potentially be used to detect the location of the robot relative to the assembly line.

Assembly Line Location

An initial estimate of the assembly line location is available based on a static map (see α_b and β_b in Fig. 2). However, the assembly line location must be within the robot's reference frame, which in turn requires an estimate of the robot's pose. Therefore, the accuracy of the initial estimate of the assembly line location relative to the robot is lower-bounded by localization accuracy.

Our system continually updates an online location estimate of the assembly line in the robot's map. We use four PX4Flow optic flow sensors [12] mounted facing downwards on the robot, as shown in Fig. 1. *Four* sensors allow for assembly line detection independent of the robot's heading or pose as it enters the line; this is of particular importance due to the omni-directional motion of the mobile base.



(a) Raw image (b) Edge detection (c) Line detection

Figure 5: The image processing pipeline to detect the assembly line location.

[2]The video is also included as a multimedia attachment with the submission, please see file: VIDEO-CLIP-1.mp4

The optic flow sensors frame the surface at a frequency of 6Hz. Using image processing techniques, our system detects whether the image includes a line corresponding to the boundary of the assembly line (see Fig. 5). Note that such a line would be present in the image when the sensor transitions from a static to dynamic surface (or vice-versa). Specifically, we used a Canny edge detector to detect edges in each image transmitted by the four sensors [13]. Next, the system calculates the Hough transform to identify lines within the detected edges [13]. Lines that differ from the expected orientation of the assembly line are eliminated from possible assembly line detection. Lastly, the robot's model of the assembly line's location is updated based on the current sensor location and the line in the image. This process mitigates error due to estimation of assembly line location based purely on localization information, and provides redundancy in assembly line detection during robot operations.

Assembly Line Speed

The optic flow sensors can also be used to sense surface velocity. We have previously demonstrated the applicability of these sensors for surface velocity detection [7]. However, this method requires an additional estimate of robot velocity to estimate conveyor belt speed using on-board optic flow sensors. Furthermore, the speed information is available only when the dynamic surface is within the field of view sensors, i.e., on or near the assembly line.

Hence, we incorporate an off-board wheel encoder mounted on the dynamic surface to measure assembly line speed. The sensing information is transmitted wirelessly to the robot. Although the presence of an off-board sensor requires additional infrastructure, the need for an estimate of the assembly line speed throughout the robot's operation demands this approach. This method does not require any additional measurement of robot velocity, alleviating the cascading effects of any error in robot velocity estimation.

Localization and Car Tracking

To achieve autonomous navigation, we additionally implement solutions for localization, tracking moving cars, and planning paths to moving goals. Localization and tracking of moving targets are closely related, as the robot must continuously keep track of its own location in the world, as well as its location relative to its moving target. Therefore, we apply an integrated approach for simultaneous localization and tracking.

Specifically, we extend the feature-based Extended Kalman Filter (EKF) localization algorithm [14], which processes geometrical features extracted from raw laser scan data (e.g., lines and corners), to correct the robot's 2D pose estimate based on odometry information. In the automotive factory setting, it is also necessary to track the 2D poses of n dynamic objects (the cars to be assembled) simultaneously. However, since the cars are fixed to a straight conveyor belt with a constant distance d between them, the task is reduced to tracking the position of the first car p_b (see Fig. 2). Given p_b, the system can simply reconstruct the pose of car i in the world frame as follows:

$$\mathbf{p}_i^W = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix}^W = \mathbf{T}_B^W \cdot \begin{bmatrix} (i-1)\cdot d + p_b \\ 0 \\ 0 \end{bmatrix}^B \tag{2}$$

$i \in \{1, \ldots, n\}$ and $\mathbf{T}_B^W$ represent the static transformation from the belt frame B into the world frame W.

For simultaneous localization and tracking, we expand the state vector of the EKF with p_B as $\mathbf{x} = \begin{bmatrix} \mathbf{x}_r^W & p_b^B \end{bmatrix}^T$, where $\mathbf{x}_r^W$ is a 2D pose estimate of the robot in the world frame.

During the prediction step of the EKF, current robot velocity measurements $\mathbf{v}_r^R$ and the belt v_b^B are used to propagate the state vector:

$$\hat{\mathbf{x}}(t) = \begin{bmatrix} \mathbf{R}(\theta_{t-1}) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{v}_r^R(t) + \mathbf{v}_r^R(t-1) \\ v_b^B(t) + v_b^B(t-1) \end{bmatrix} \frac{\Delta t}{2} + \mathbf{x}(t-1) \tag{3}$$

$\mathbf{R}(\theta_{t-1})$ is the rotation matrix and Δt represents the time elapsed since the last update. To account for surface motion, the odometry information from the wheel encoders must be rectified with the sensed surface velocity as follows:

$$\mathbf{v}_r^R = \mathbf{v}_{\text{odom}}^R + \mathbf{v}_{\text{surf}}^R \tag{4}$$

During the measurement update, first an association step is carried out using a nearest neighbor search [15], and then new feature observations are processed in order to correct the predicted state. In general, this includes feature observations from the static environment f_{stat} and dynamic features f_{dyn} originating from moving objects. For static features, this approach follows a standard update routine as described by Thrun et al. [16], resulting in a correction of $\mathbf{x}_r^W$. In the case of dynamic features, the measurement model is defined as a function h transforming a mapped feature k of dynamic object i $\mathbf{m}_k^I$ into the robot frame using the objects current pose estimate from Eq. 2:

$$f_{dyn,k} : \mathbf{z}_k^R = h\left(\mathbf{x}_r^W, p_b^B, \mathbf{m}_k^I\right) + \mathcal{N}(0, \mathbf{R}) \tag{5}$$

Noise is assumed to be Gaussian, and $\mathbf{R}$ represents the measurement noise covariance matrix. The dynamic features are mapped a priori relative the object's coordinate system. As can be determined from Eq. 5, the transformation of mapped feature k depends upon both the robot's pose and p_B; thus, the complete state vector is updated when the system observes a dynamic feature.

Path Planning

The path planning module computes safe, smooth paths (which serve as input to the trajectory tracking algorithm) to potentially moving targets, while avoiding dynamic obstacles such as humans. We adopt a standard planning architecture, wherein a global planner (A*) generates a path to the goal; this path is then optimized at runtime using a local planner (an implementation based on the Elastic Band planner [17]). The local path planner allows for quick reaction to unforeseen obstacles, as it incorporates previously computed solutions to generate optimal paths based on current sensor data. These properties are desirable in the dynamic, human-centric environment of a factory floor. Further, the ability to efficiently incorporate prior solutions into closed-loop path optimization at runtime is of critical importance when planning paths to dynamic goals (e.g., for approaching cars on a moving conveyor belt or for handling perturbations in robot localization when transitioning onto and off of the belt).

System Evaluation on Factory Floor

We conducted our system evaluation on an operational automotive final assembly line. Each car spent roughly 150-180 seconds on the line due to conveyor belt motion within the robot's workspace. The assembly line, while in its "on" state, moved at an average speed of 78.9 mm/s ($\sigma = 1.7$ mm/s) as measured by the off-board contact sensor. The workstation included human workers who were typically assembling cars adjacent to the car being worked on by the robot, but who did occasionally work simultaneously with the robot on the same car. The robot detected the surrounding humans using its on-board laser scanners, and used this information to plan collision free paths using the path planning module.

The robot's task involved movement between the following waypoints: Start (on a static surface), Goal-A (on a conveyor belt), and Start. The robot performed a proprietary assembly task at Goal-A; this task had to be completed within the cycle time for which the car was present at the robot's workstation. To evaluate the system's performance within a more challenging scenario, we also had the robot complete assembly tasks at both Goal-A and Goal-B within a single time cycle. We collected data logs from 15 iterations of the test scenario, including one iteration of the task involving Goal-B. The robot took an average of 54.88 s ($\sigma = 3.34$s) to complete the single-sided task. The average navigation time was 19.83 s ($\sigma = 2.87$s).

Achievement of Key Behaviors

The robot transitioned between static and dynamic surfaces twice during each task. The sensing sub-system updated the assembly line location a median of 1.5 times during each transition; the robot detected this transition at least once per trial. The assembly line speed was detected through the off-board, contact-based sensor (see Fig. 6a). Assembly line speed was available to the robot throughout its motion at an update rate of 30 Hz.

We validated the control algorithm using a position hold task with robot positioned partially on and partially off of the dynamic surface (see Fig. 6b); a video of this test is available[3] at `http://tiny.cc/mra2` . This scenario was particularly challenging due to the excessive, damaging torques the robot could experience in the event that the control algorithm did not compensate for surface motion. The robot successfully held its position during the task despite being only partially positioned on the moving conveyor belt. Note that when the desired velocity was zero, the wheel commands were non-zero (see Fig. 7). For this trial, the mean deviation in robot position was < 2 cm even as the robot was positioned partially on and partially off of the assembly line. We additionally tested the system by teleoperating it on the dynamic surface; the results of these tests are depicted in an attached video[4] (see `http://tiny.cc/mra3`).

System Performance

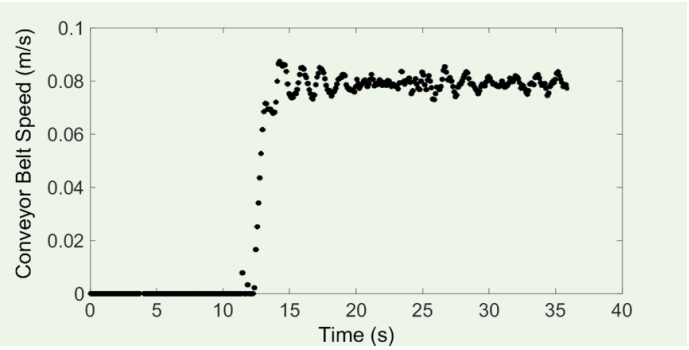
Lastly, we evaluated the performance of our autonomous robotic system during a time-critical task on an operational line. Figure 8a depicts the robot's path during one prototypical run, demonstrating the integrated performance of the system. Similarly, Fig. 9 depicts the trial in which the robot performed its assembly task on either side of the moving vehicle. A video of the robot's autonomous navigation from static to dynamic surface and vice-versa is available[5] at `http://tiny.cc/mra4a` and `http://tiny.cc/mra4b`. The robot was able to successfully follow desired velocity commands during its task (Fig. 8b). The ability to track the desired robot velocities demonstrates successful operation of both the control and sensing sub-systems.

The dynamic surface changed its state periodically throughout the trials: e.g., the assembly line (and, consequently, the car) occasionally stopped moving after the robot began its motion, due to normal operations within the factory. By maintaining an estimate of the assembly line speed and location, the robot was able to accomplish its task despite unscheduled changes in the state of the assembly line.

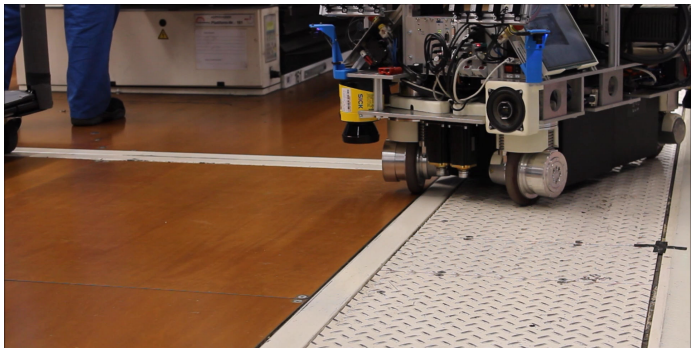
[3]The video is also included as a multimedia attachment with the submission, please see file: VIDEO-CLIP-2.mp4

[4]The video is also included as a multimedia attachment with the submission, see file: VIDEO-CLIP-3.mp4

[5]The video is also included as a multimedia attachment with the submission, please see file: VIDEO-CLIP-4a.mp4 and VIDEO-CLIP-4b.mp4



(a) A typical speed profile of the assembly line as it transitions from an "off" to "on" state.



(b) The mobile robot performing a position hold while positioned partially on and partially off of the assembly line.

Figure 6: The motion of the conveyor belt and an example of its effect on our mobile robot system.

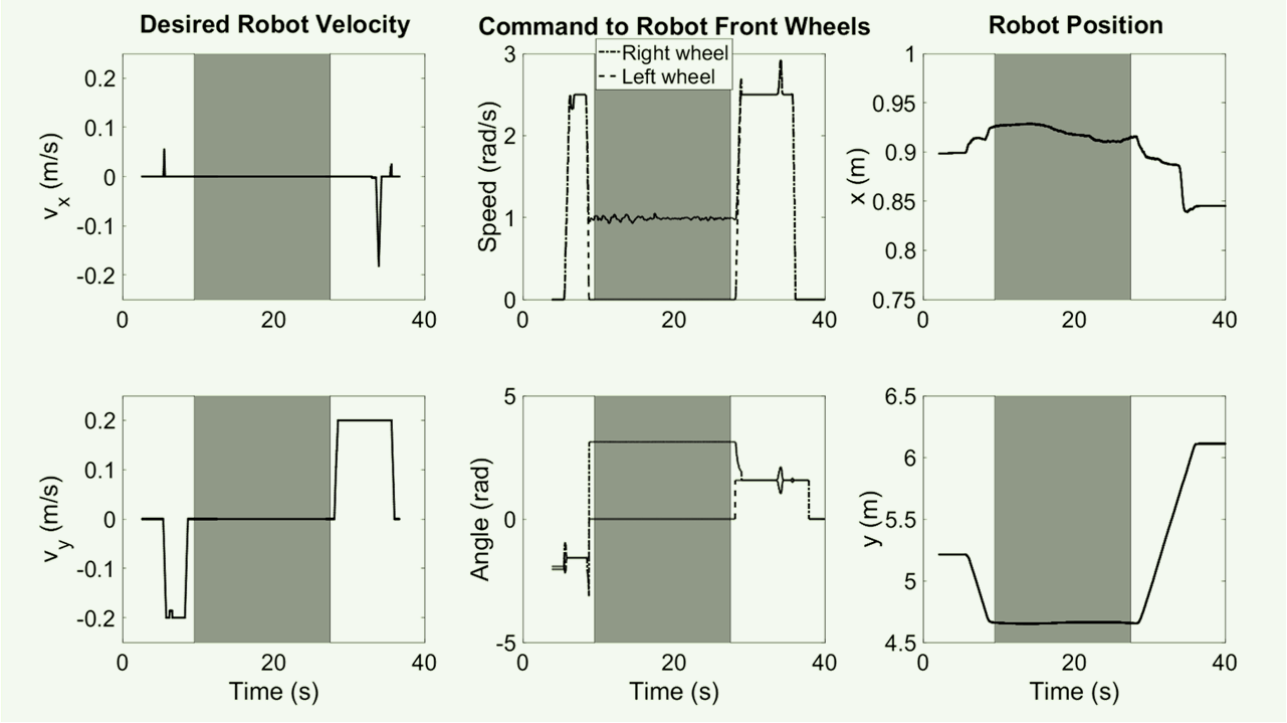
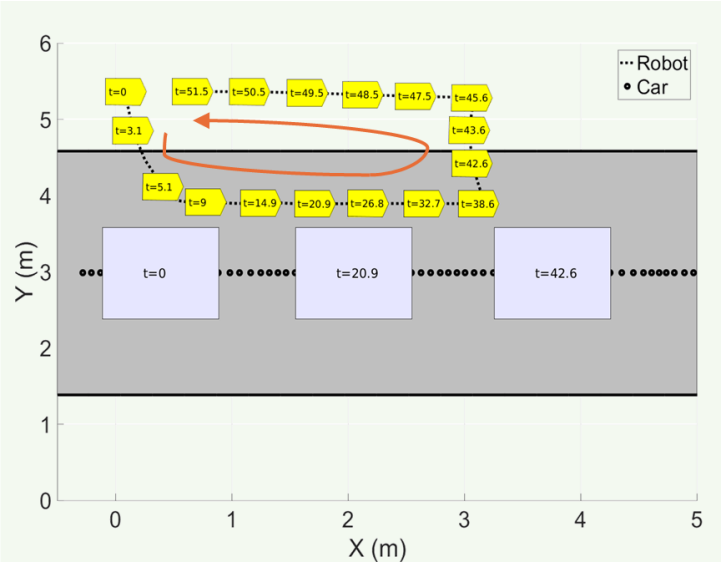
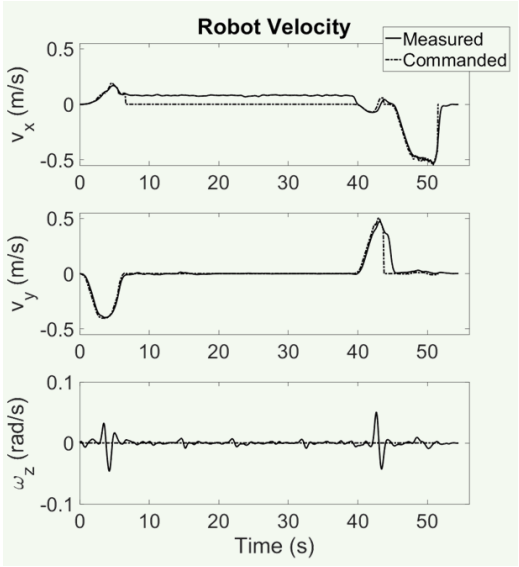


Figure 7: The operation of the control sub-system during the position hold task. (left) The desired linear velocity of the robot was provided by a human operator. The desired angular velocity was zero throughout the scenario, and is not included in the figure. The shaded region denotes the duration for which the desired robot velocity was zero (i.e., when the robot was required to hold its position). The plots in the center depict the commands issued to the front wheels of the robot. The commands to the front right wheel were non-zero during the position hold task in order to compensate for the motion of the assembly line. The plots on the right depict the robot's pose during the task. Despite the robot's position being partially on and partially off of the moving assembly line, we observed only minimal deviations to robot pose when the commanded velocity was zero.



(a) A typical robot trajectory for the test scenario, with the assembly task performed only on the left side of a moving car (Goal-A, as depicted in Fig. 2). The gray area denotes the dynamic surface, the yellow polygon indicates the robot, and the blue rectangle represents the moving car on the conveyor belt. The orange arrow denotes the direction of robot motion. Time stamps for the robot and car are provided in the figure. Once the robot is positioned next to the car (Goal-A), it remains stationary relative to the car; this is observed as the linear motion of the robot along the assembly line.



(b) The robot's absolute velocity for the trial depicted in Fig. 8a. The commanded velocity is autonomously generated by the robot's motion planner. Note that commanded velocity is zero when the robot is positioned at Goal-A. The measured robot velocity is obtained by post-processing the estimate of the robot's pose from the localization algorithm. The system successfully tracked the velocity commands during the demonstration.

Figure 8: The performance of the integrated system during the time-critical factory task.

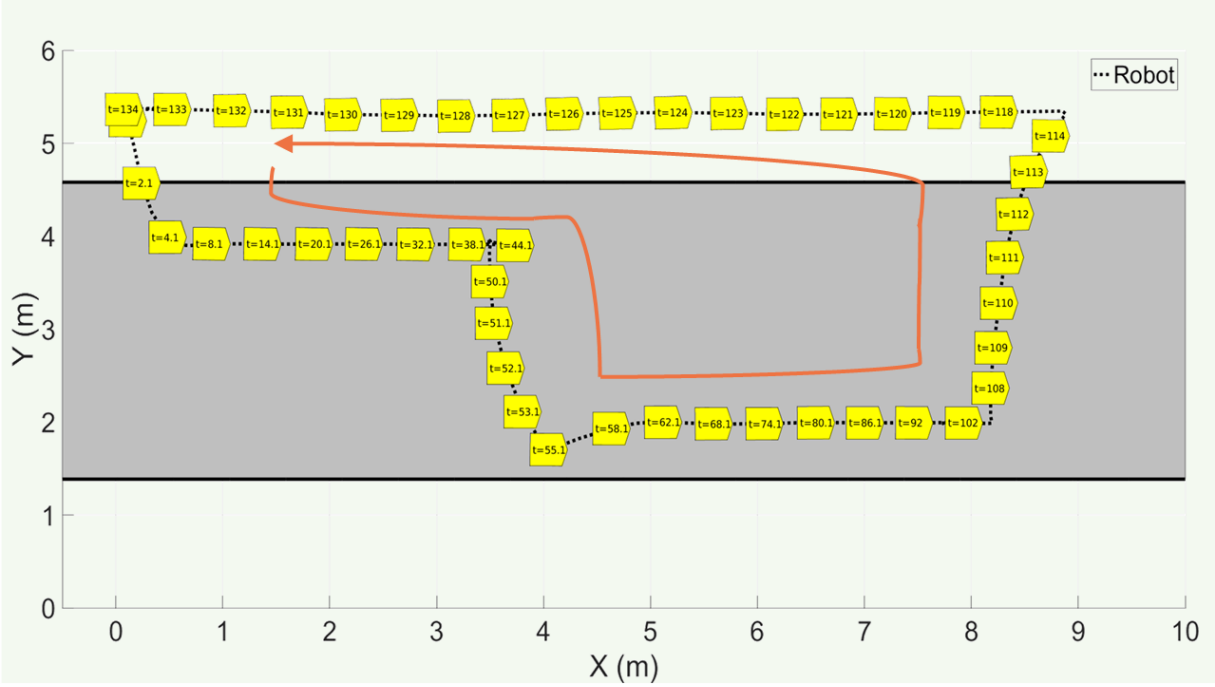


Figure 9: The robot trajectory for the test scenario during which the assembly task was performed on both sides of the car (Goal-A and Goal-B). The orange arrow denotes the direction of robot motion. This task required the robot to travel farther and spend more time on the dynamic surface than the single-side task.

Implications for Human-Robot Collaboration in Automotive Final Assembly

Equipped with a capability for autonomously navigating on dynamic surfaces, our mobile robot can perform tasks such as delivering parts to human associates on the conveyor belt and carrying out assembly tasks on moving cars. A freely mobile system increases the effective operating region, as compared to a robot with constrained mobility, provides greater flexibility when designing automotive manufacturing processes, and can achieve improvements in task efficiencies while sharing its environment with humans.

In order to highlight this benefit, we conducted a simulation evaluation in which the system predicts human motion and plans to navigate amongst the human in a shared workspace with a static surface. The evaluation compared the performance of a freely moving robot to a constrained robot whose mobility was limited to a linear axis [18]. Similar to [18], we simulated human motion by recording in a motion capture system the trajectories of real humans walking to four goal locations. We drew from the dataset a random sequence of eight recorded trajectories for use in each simulation task. The robot's task was to navigate to a sequence of goal locations while sharing the environment with the simulated human. Both the freely moving and constrained robots used MPS [19] a system for predicting human motions and SIPP [20] a planner for generating robot paths using time-indexed predictions. A safety stop was triggered during task execution if the distance between human and the robot reduced beyond a safety threshold. Similarly, the motion of the simulated human was paused if the robot was in her way. The two robots used a different set of motion primitives due to the difference in their degrees of freedom; the rest of the simulation parameters were set identically for the two robots.

We ran ten simulation trials of the task and observed that the freely moving robot required substantially less time on average to complete its task (147.9s), as compared to the constrained robot (167.3s). Further, the human required less time on average to complete its sequence of eight motions while sharing the environment with the freely moving robot (160.3s), as compared to the constrained robot (167.0s). Both these differences were statistically significant ($p < 0.05$, Wilcoxon signed rank test), and were realized using the additional degree of freedom available to the freely moving robot to adapt and plan around the human. Lastly, the simulations indicated a higher number of safety stop triggers on average for the freely moving robot (2.1) as compared to the robot limited to a linear axis (1.1).

These simulations provide proof of concept that performance benefits can be realized with a freely moving robot working alongside humans in automotive final assembly. However, additional challenges must be addressed to field collaborative mobile robots on an AFAL, including consideration of safety (see ISO 10218-2 and ISO/TS 15066) and maintainability. In addition, techniques for predicting human motion and conveying robot intent will be required to achieve anticipatory robot behavior and realize the benefits in task performance.

The modular design of our control and sensing system will enable the development and testing of algorithms for human-robot collaboration without additional consideration of dynamic surfaces, and will facilitate the integration of these technologies with a system that can effectively navigate on moving-floor assembly lines. We believe the autonomous navigation capability presented herein is a necessary first step towards new forms of human-robot collaboration in automotive final assembly and will provide greater flexibility in planning next-generation manufacturing processes.

References

[1] J. Krüger, T. Lien, and A. Verl, "Cooperation of human and machines in assembly lines," *CIRP Annals-Manufacturing Tech.*, vol. 58, no. 2, pp. 628–646, 2009.

[2] V. Krueger, A. Chazoule, M. Crosby, A. Lasnier, M. R. Pedersen, F. Rovida, L. Nalpantidis, R. Petrick, C. Toscano, and G. Veiga, "A vertical and cyber–physical integration of cognitive robots in manufacturing," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1114–1127, 2016.

[3] R. D. Schraft, C. Meyer, C. Parlitz, and E. Helms, "Powermate-a safe and intuitive robot assistant for handling and assembly tasks," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 4074–4079, IEEE, 2005.

[4] R. Müller, M. Vette, and M. Scholer, "Robot workmate: A trustworthy coworker for the continuous automotive assembly line and its implementation," *Procedia CIRP*, vol. 44, pp. 263–268, 2016.

[5] J. Shi and R. Menassa, "Flexible robotic assembly in dynamic environments," in *Performance Metrics for Intelligent Systems Workshop*, pp. 271–276, ACM, 2010.

[6] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, C. Fantuzzi, and K. Fuerstenberg, "Advanced sensing and control techniques for multi AGV systems in shared industrial environments," in *Emerging Technologies & Factory Automation (ETFA)*, pp. 1–7, IEEE, 2015.

[7] V. V. Unhelkar, J. Perez, J. C. Boerkoel Jr, J. Bix, S. Bartscher, and J. A. Shah, "Towards control and sensing for an autonomous mobile robotic assistant navigating assembly lines," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 4161–4167, 2014.

[8] Harvard Business Review, "Smarter smaller safer robots." `hbr.org/2015/11/smarter-smaller-safer-robots`. [Online; last accessed August 2017.].

[9] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation," *International Journal of Robotics Research (IJRR)*, vol. 34, no. 3, pp. 335–356, 2015.

[10] C. Connette, A. Pott, M. Hagele, and A. Verl, "Control of an pseudo-omnidirectional, non-holonomic, mobile robot based on an ICM representation in spherical coordinates," in *Conf. on Decision and Control (CDC)*, pp. 4976–4983, IEEE, 2008.

[11] S. Lee and J.-B. Song, "Robust mobile robot localization using optical flow sensors and encoders," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 1039–1044, IEEE, 2004.

[12] D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys, "An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 1736–1741, 2013.

[13] D. Sundararajan, *Digital Image Processing: A Signal Processing and Algorithmic Approach*. Springer, 2017.

[14] F. Mirus, F. Slomian, S. Doerr, F. Garcia Lopez, M. Gruhler, and J. Pfadt, "A modular hybrid localization approach for mobile robots combining local grid maps and natural landmarks," in *Symposium on Applied Computing*, (Pisa, Italy), ACM, 2016.

[15] P. Konstantinova, A. Udvarev, and T. Semerdjiev, "A study of a target tracking algorithm using global nearest neighbor approach," in *Intl. Conf. on Computer Systems and Technologies*, pp. 290–295, 2003.

[16] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Boston, Massachussetts: MIT Press, 2005.

[17] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 802–807, IEEE, 1993.

[18] V. V. Unhelkar*, P. Lasota*, Q. Tyroller, R.-D. Buhai, M. Laurie, B. Deml, and J. A. Shah, "Human-Aware Robotic Assistant for Collaborative Assembly: Integrating Human Motion Prediction with Planning in Time," *IEEE Robotics and Automation Letters (RA-L)*, 2018.

[19] P. A. Lasota and J. A. Shah, "A multiple-predictor approach to human motion prediction," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 2300–2307, IEEE, 2017.

[20] M. Phillips and M. Likhachev, "SIPP: Safe interval path planning for dynamic environments," in *Intl. Conf. on Robotics and Automation (ICRA)*, pp. 5628–5635, IEEE, 2011.