2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Towards Control and Sensing for an autonomous Mobile Robotic Assistant navigating Assembly Lines

Vaibhav V. Unhelkar*[1], Jorge Perez[1], James C. Boerkoel Jr.[2], Johannes Bix[3], Stefan Bartscher[3] and Julie A. Shah[1]

***Abstract*— There exists an increasing demand to incorporate mobile interactive robots to assist humans in repetitive, non-value added tasks in the manufacturing domain. Our aim is to develop a mobile robotic assistant for fetch-and-deliver tasks in human-oriented assembly line environments. Assembly lines present a niche yet novel challenge for mobile robots; the robot must precisely control its position on a surface which may be either stationary, moving, or split (e.g. in the case that the robot straddles the moving assembly line and remains partially on the stationary surface).**

In this paper we present a control and sensing solution for a mobile robotic assistant as it traverses a moving-floor assembly line. Solutions readily exist for control of wheeled mobile robots on static surfaces; we build on the open-source Robot Operating System (ROS) software architecture and generalize the algorithms for the moving line environment. Off-the-shelf sensors and localization algorithms are explored to sense the moving surface, and a customized solution is presented using PX4Flow optic flow sensors and a laser scanner-based localization algorithm. Validation of the control and sensing system is carried out both in simulation and in hardware experiments on a customized treadmill. Initial demonstrations of the hardware system yield promising results; the robot successfully maintains its position while on, and while straddling, the moving line.

I. Introduction

To cater to the needs of modern factories, industrial robots are increasingly becoming more mobile [1], [2], adaptive [3] and human-oriented [4], [5]. We are removing the cages that separate them from people and are now deploying them into human-centric factory environments. Some assembly work, such as automotive final assembly, involves complex and highly dexterous tasks that still require human capabilities. However, near-term opportunities exist to improve the productivity of people in performing "value-added work" by deploying robots to perform the "non-value-added work."

This paper explores one such opportunity, which is to field a mobile robotic assistant that travels across a factory, delivering tools and materials to the human workers. Assembly manufacturing facilities, such as in the automotive and aerospace sectors, tend to be quite large in area and often require multiple deliveries of parts over long distances and at regular intervals. An adept mobile robotic assistant can provide a significant boost to human worker productivity by transporting these items, thereby freeing the worker to focus on the tasks that require their human capabilities. Ideally the introduction of mobile robotic assistants would require minimal modifications to the factory layout and infrastructure, thus preserving operational flexibility as compared to fixed infrastructure solutions.

There is a breadth of prior work aimed at designing and deploying robots as assistants, for example in health-care [6]–[8], in recreational environments such as museums and shopping malls [9]–[11], and in the home [12], [13]. In this work, we aim to develop a mobile robotic assistant to perform fetch-and-deliver tasks in a human-oriented assembly line environment. Successful deployment of a mobile robotic assistant on the factory floor requires mechanisms (both hardware and software) to ensure the safety of the human associates. To provide benefit, the robotic assistant requires navigation, path planning, deliberative task planning/scheduling, and human interface capabilities, ensuring delivery of the right materials to the point-of-use at the right time [14], [15]. However, prior to accomplishing any task on the assembly line, the robot first needs to be able to traverse a moving conveyor belt (see Fig. 1). This requires that it precisely control its position on a surface which may be either stationary, moving, or split (e.g. in the case that the robot straddles the moving assembly line and remains partially on the stationary surface).

In this paper, we describe a solution for the autonomous control of the mobile robotic assistant as it traverses a moving-floor assembly line. The Rob@Work 3 mobile base [16] is used as the basic platform to develop and demonstrate these custom capabilities. The Rob@Work 3 mobile robot

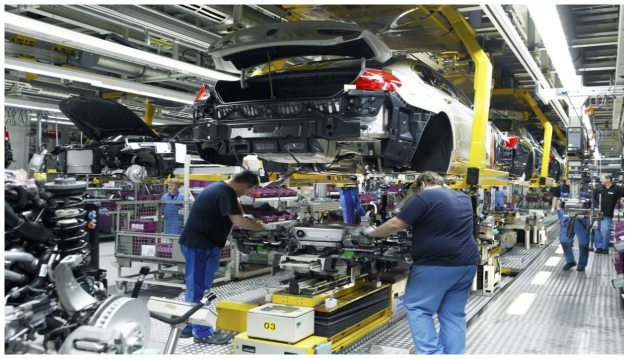

Fig. 1. A typical automotive assembly line with a moving surface. (Photo Credits: https://www.press.bmwgroup.com/)

*Corresponding author: unhelkar@mit.edu

[1]V. V. Unhelkar, J. Perez and J. A. Shah are with the Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge, Massachusetts,

[2]J. C. Boerkoel Jr. is with the Computer Science Department, Harvey Mudd College, California,

[3]J. Bix and S. Bartscher are with the Innovation Production Division, BMW Group, Munich, Germany.

supports navigation in static environments, and comes with both proprietary and open-source solutions for its basic operation. In this work we build on the open-source Robot Operating System (ROS) [17] software architecture and generalize the algorithms for the moving line environment. Wherever possible, off-the-shelf sensors and algorithms are customized and integrated for the application at hand.

We begin by describing the Rob@Work 3 robot platform, its hardware and software capabilities, as well as introduce the additional sensors necessary to perform navigation and control on the assembly line. Next in Section III we describe the control architecture and the modifications that enable reaction to sensed belt rates. Sensing of the moving surface and the robot state is a prerequisite for successful implementation of the controller; hence, possible sensing alternatives along with details of implementation are presented in Section IV. Lastly, the developed system is tested via both simulation and hardware (Sections V and VI). Gazebo robot simulator [18] is first used to validate the control system in simulation. The control and sensing subsystem is implemented on robot hardware, and first validation with off-board sensing is conducted on an analogue assembly line using a customized treadmill. The paper concludes with a discussion of the performance of the current system, its limitations and possible directions for further improvement.

II. Rob@Work 3: Mobile Robotic Base

Rob@Work 3 is a mobile robotic platform, developed by Fraunhofer IPA®, for industrial applications (see Fig. 2). For its motion the mobile base uses four independently actuated wheels; each wheel has two motors providing degrees of freedom in steering (via steering angle, ψ) and driving (via angular rate, $\dot{\theta}$). Sensing is achieved by encoders at each of the eight wheel motors and two SICK S300 laser scanners[1]. The odometry information is obtained using the eight encoders, which is then combined with the laser scanners to perform localization and mapping.

Robot Operating System (ROS) is used as the middleware for the robotic platform[2]. Safety features, at both the software and hardware level, are pre-built into the robot thus providing the basic measures for ensuring its safe operation in human environments. The ROS middleware also provides easy access to the open-source library of various robotics algorithms, which can be used off-the-shelf or customized for the mobile base. The existing software of Rob@Work 3 is capable of motion planning and navigation in static domains using the aforesaid sensors, actuators and widely used robotics algorithms for localization [19], mapping [20] and navigation [21]. Table I provides the salient features of the Rob@Work 3 platform.

A. *Problem Description: Navigating Assembly Lines*

The basic robotic platform, through the aforementioned features, can capably navigate in static, known environments. However, navigating an assembly line *additionally* requires

[1]`http://goo.gl/VwoA42`
[2]`http://wiki.ros.org/care-o-bot`

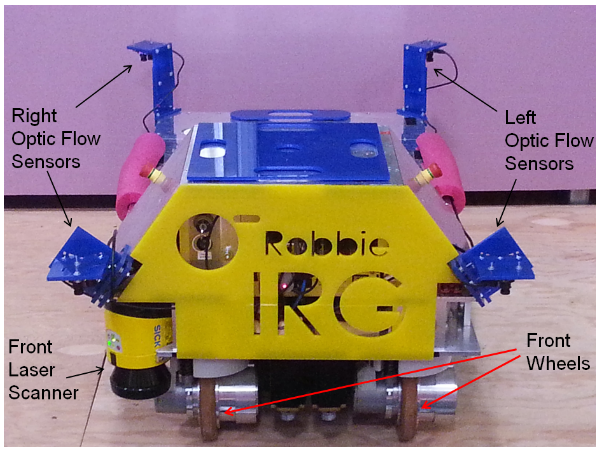

Fig. 2. The modified Rob@Work 3 mobile robotic platform, with four optic flow sensors for augmented surface sensing capabilities.

the robot to control its location and velocity on a moving surface. At any time, the robot may be entirely on, entirely off, or else straddling the moving surface. These unconventional but practical scenarios require additional sensing and a customized control algorithm for the robotic mobile base.

In this work we assume the location of the assembly line in the factory is known and static. A number of exogenous factors influence the stopping and starting of the assembly line. Factories oriented around manual work do not typically maintain a reliable signal of the lines' *on/off* state and thus this state must be sensed directly by the robot. However, when *on*, the assembly line moves at an almost constant velocity ($\pm 10\%$ variation). Other objects in the factory (part carts, cars, people) will move over different timescales and the robot will need localization and path planning algorithms that are suitable for dynamic environments. In this initial effort we assume these objects are static and focus on the problem of control and sensing on the assembly line belt. Our sensing and control sub-systems are designed to be modular, so that they can be coupled with a variety of localization, mapping and navigation algorithms.

Augmentation of Hardware: We augment the robot mobile base by installing a suite of four PX4Flow optic flow sensors [22] to provide the additional sensing of the state of the moving line. This augmented system, along with its selection, technical details, and performance, is presented next.

TABLE I
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

B. Reference Frames

The robot, due to its numerous parts and sensors, requires maintenance of over fifteen reference frames for its complete kinematic description. Primarily, three reference frames are used in the following sections: map, robot, and optic$_i$. The map frame, denoted by superscript M and used as reference for localization, is the static frame fixed to a reference point in the factory. The robot frame, denoted by superscript R, is fixed to the robot body, and is used for maintaining the robot odometry and location with respect to the static world frame. Finally, a reference frame is defined for each of the four optic flow sensors. This is used to suitably transform the sensor information into the robot frame.

III. Control

The control architecture of the mobile base uses multiple feedback loops [23]. A path planning algorithm (or a human teleoperator) issues a desired trajectory, which is translated into velocity commands for the mobile robot. The desired velocity of the ith 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,

$$v_{\text{x,i}} = \dot{x}_\text{r} - \dot{\phi}_\text{r} y_{\text{w,i}} \tag{1a}$$
$$v_{\text{y,i}} = \dot{y}_\text{r} + \dot{\phi}_\text{r} x_{\text{w,i}}. \tag{1b}$$

Each wheel velocity command is then converted to the wheel configuration, a steering angle ψ and angular rate $\dot{\theta}$ command, as detailed in [23]. The existing PD controller is used for controlling ψ and $\dot{\theta}$ for each wheel. The control architecture, implemented in ROS, incorporates basic safety considerations that are essential while operating in an industrial environment. We modify the existing wheel controller, rather than design anew, to preserve these in-built safety features.

The nominal wheel controllers, described above, operate assuming the commands are issued for the case of static surface. Hence, if any of the wheels are on a moving surface, the nominal controller will not provide the desired response. To overcome this issue, we compensate the command for each wheel ($\mathbf{v}^R_{\text{wheel,i}}$) based on the absolute surface velocity at its point of contact ($\mathbf{v}^R_{\text{surf,i}}$). Algorithm 1 describes how the commanded velocity for each wheel is altered assuming the knowledge of surface velocity. The modified wheel velocity command is used to compute the desired wheel configuration (ψ,$\dot{\theta}$). This modification preserves use of the existing software architecture and wheel controller.

```
Input: Nominal command to the wheel controller
Output: Compensated command to the wheel controller
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
```

Algorithm 1: Modification to the nominal wheel controller command for navigating Assembly Lines

IV. Perception

As described in Sec. III, information about surface velocity is essential for the modified controller. Further, an alternate method is required for localization since odometry degrades when the robot is navigating on moving surfaces.

A. Sensing Surface Velocity

Sensing surface velocity requires additional hardware for the robotic platform. A number of different sensing options exist for velocity estimation of surrounding objects. For instance, various types of radars are used to measure velocities of cars, or to detect aerial objects. Typically robotic systems use vision (both stereo and rgbd) algorithms to estimate velocity of surrounding objects. In our application, the robot is always in contact with the surface, and therefore contact sensors, too, may be used for detecting surface velocity. Surface motion can also be detected indirectly by comparing two independent sources of localization, one linked to the map reference frame (such as, inertial sensors) while the other linked to the local robot frame (such as, odometry).

Sensing Alternatives: Constrained by size and power requirements, we explored the use of four different sensors for measuring surface velocity: miniature radars [24], optic flow sensors [22], contact-based encoders, and inertial sensors. As the surface moves relatively slowly (usually < 0.2 m/s), the performance of miniature radars and low-cost inertial sensors is limited by accuracies at low speeds. Further, an indirect method (such as, inertial sensor based system) will be reactive, detecting surface motion only via disturbance in the robot's motion caused by the surface. An on-board contact sensor will negatively affect the ground clearance of the robot, limiting its mobility over uneven surfaces. Hence, in our work, we evaluate the use optic flow sensors to detect surface velocity.

Sensing via Optic Flow: The PX4Flow is a CMOS image based optical sensor designed for mobile robotic applications, including aerial vehicles [22]. The sensor includes a microcontroller, image sensor, gyroscope and sonar with on-board processing, and interfaces with the robot through ROS[3]. The image gradient is processed on-board using a Kalman filter implemented on the microcontroller, to obtain a refined estimate. The derived optic flow is next compensated for image distance (using sonar), and sensor rotation (using on-board gyroscope) to provide the image velocity.

For our application, PX4Flow sensors are mounted facing downwards and need to be augmented with a light source to measure the relative velocity of the surface. As described in Sec. II, the assembly line operates in only two states and so the requirement of detecting surface velocity reduces to that of detecting surface motion. Use of optic flow for this purpose requires that the surface to have sufficient image features, and we empirically validate that this is the case for our particular factory environments. The sensors are mounted at a fixed height and maintain a constant distance from the surface. Hence, the fixed height of the sensor is

[3]`http://wiki.ros.org/px4flow_node`

used in calculating velocity, rather than the noisy sonar measurements. This modification considerably reduces the noise in the velocity measurement for our application. We then process the output velocity using a low pass filter before the signal is used in the control algorithm.

The sensors are attached to the robot frame to measure surface velocity relative to the robot at the ith wheel ($\mathbf{v}^{\text{optic}_i}_{\text{surf}_i\text{-robot}}$), which is then transformed from sensor frame, optic$_i$, to the robot frame, R, to provide $\mathbf{v}^{R}_{\text{surf}_i\text{-robot}}$. As the controller requires absolute (and not relative) surface velocity, we need additional information about the robot's state. Sec. IV-B provides details of the algorithm used to obtain the robot's absolute velocity, which is then combined with the sensor measurement to calculate the absolute surface velocity,

$$\mathbf{v}^{R}_{\text{sensor}_i} \triangleq \mathbf{v}^{R}_{\text{surf}_i\text{-robot}} = \mathbf{v}^{R}_{\text{surf}_i} - \mathbf{v}^{R}_{\text{robot}} \tag{2a}$$

$$\therefore \mathbf{v}^{R}_{\text{surf}_i} = \mathbf{v}^{R}_{\text{sensor}_i} + \mathbf{v}^{R}_{\text{robot}}. \tag{2b}$$

Sensor Positioning: A suite of four PX4Flow optic flow sensors, located at robot's corners, is integrated with the Rob@Work 3 for detecting surface velocity (Fig. 2). As the robot is omni-directional and can be at an arbitrary relative orientation with respect to the moving line, we need a sensor at each of the four wheels. Due to mechanical limitations, the sensor cannot be placed at the wheel's location and thus are positioned at the vertices of the robot's rectangular planform. Localization information is used to detect when the wheels are on the assembly line, and the optic flow sensors are used to detect the surface motion. This solution ensures detection of the moving surface for each wheel and arbitrary robot orientation, since the robot planform is convex in shape.

B. Robot Localization

The robotic platform in its nominal operation uses the wheel encoders and laser scanner data to obtain its position. However, when the robot is on a moving surface the encoder information degrades and cannot be used to reliably estimate the robot's position. Further, when the robot is completely on the moving surface, the base will move along with the surface in spite of the wheels being stationary. This causes inconsistencies between the laser scanner and encoder data, and results in inaccurate localization using algorithms that rely on wheel encoders. As a result, there is a need to estimate the robot state without the use of wheel encoders once the robot enters a moving surface.

Rather than using wheel encoders, localization can be performed with a suite of 3-axis accelerometers and gyroscopes. However, performance of inertial sensors degrades with time and must be corrected with additional input, such as GPS or vision-based localization. We pursue an approach that uses the robot's existing onboard sensors, and that does not require external infrastructure (as an indoor positioning system like solution does). Specifically we explore the application of Laser scanner based localization algorithms.

Laser-only based localization: Robot localization using only laser scanner measurements is a problem that has been studied in detail [25], [26]. Successful algorithms have been tested in real systems and have been developed and implemented. For the mobile robotic assistant, we use a scan matching approach to localization as described in [26], primarily because of its open source ROS implementation[4]. Further, the algorithm requires low computational resources and is capable of simultaneous localization and mapping. Implementation of the algorithm on our robotic hardware results in acceptable localization performance in static environments.

Rectifying odometry: Laser scanner-only based localization provides absolute robot velocity even when the robot is on a moving surface. In contrast, the robot's motion due to the moving surface can result in misleading information from wheel encoders. Equation 3 compensates the velocity information obtained via wheel encoders; this compensated velocity can be used as an input for calculating odometry (e.g. for dead-reckoning), thereby mitigating the error in the robot's odometry on moving surfaces.

$$\mathbf{v}^{R}_{\text{encoder}_i,\text{rectified}} = \mathbf{v}^{R}_{\text{encoder}_i} + \mathbf{v}^{R}_{\text{surf}} \tag{3}$$

The above correction further allows for use of the nominal localization algorithm when the robot is not on the moving surface. This is desirable since information from the wheel encoders is still reliable when the robot is fully on a stationary surface.

V. Experimental Testing

We perform initial validation of the controls and sensing subsystems using both software simulation and hardware tests. Fig. 3 provides pictures of the two test environments.

Gazebo Simulator: The Gazebo simulator [18] is the natural choice for software validation in simulation since the robot uses ROS as its middleware, and Gazebo allows for testing of the hardware-ready code in simulation. Tests are conducted in a simulated factory-like environment with a moving line.

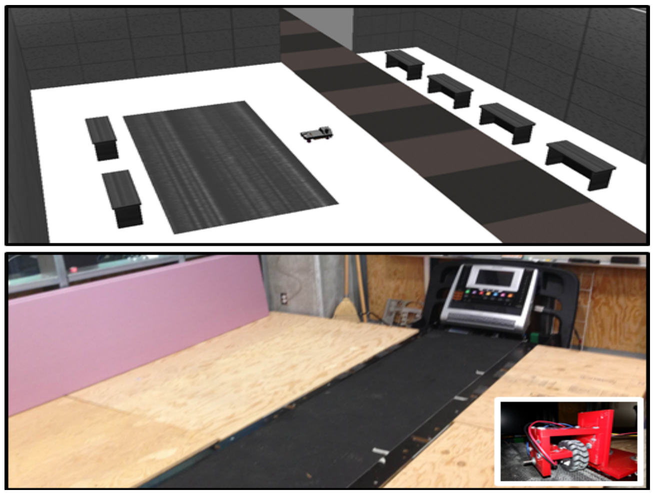

Fig. 3. Test environments for validating the designed control and sensing sub-systems: the Gazebo software simulation (top) and treadmill-based hardware test setup (bottom). Ground truth sensor, shown in the right bottom corner, is mounted on the test setup.

[4]`http://wiki.ros.org/hector_slam`

Customized Treadmill: We use a customized treadmill[5] to test the system on a real moving surface. The velocity of the treadmill can be controlled, thereby enabling testing at different operating conditions.

True Surface Velocity: A wheel-encoder based contact sensor is used to obtain the ground truth information about the surface velocity (in this case velocity of the treadmill). This ground truth sensor is mounted independently of the robot, and measures the angular displacement of a free wheel[6] in contact with the treadmill. The angular displacement of the free wheel is measured using a quadrature encoder, and sufficient friction exists between the free wheel and the treadmill surface to prevent wheel slippage and ensure reliable sensing. A software package using the Arduino micro-controller and ROS interface has been designed to collect and record information from this sensor.

VI. Results

This section presents the results from simulation and hardware tests using the proposed control and sensing algorithms.

A. Sensing Surface Velocity with Optical Flow

The performance of the PX4Flow sensor is first compared to the ground truth information, to gauge its accuracy and inform the requirement of additional processing of the raw measurements. Experimentation uncovers variation in optic flow sensor performance due to lighting conditions. Hence, the optic flow sensor is augmented with an additional light source to ensure its successful operation. Fig. 4 shows the performance of the optic flow sensor for a challenging practical scenario: where the moving surface starts from rest and then reaches its maximum velocity, and then again resumes its original rest state.

As can be observed from the Fig. 4, the sensor is capable of tracking the surface motion albeit with substantial noise. Further we observe that performance and measurement range of the optic flow sensor is dependent on the quality of image features present on the surface. High levels of noise in surface sensing will produce cascading effects in the controller performance, and rapidly changing commands will negatively affect the actuator hardware. This motivates the design of a discrete time Kalman filter with its state as the surface velocity. The process model accounts for the variation in the state through a discrete white noise term with variance σ^2. Process noise is characterized based on the variation in the surface speed during its on state, which for our application is approximated as $3\sigma = 1$ cm/s. The measurement noise is also assumed to be white for filter design, and its standard deviation is approximated as 3 cm/s, based on the in-house tests. A Kalman filter with steady-state gains is used rather than one with time varying gains. This is equivalent to a low pass filter with gains set to the steady-state gains of the Kalman filter.

B. Gazebo simulation

To validate the modifications to the control algorithm prior to its implementation on the robot hardware, we first use the test environment developed in the Gazebo simulator (Sec. V). To validate the controller independently of the sensing, the Gazebo simulation assumes perfect sensing of the surface velocity. The robot's task is to navigate across the moving assembly line. An off-the-shelf path planner [21] is used to plan the robot's path. Fig. 5 shows the deviation of robot's center from the desired path. Using the modified system, the maximum deviation is observed to be <4cm for this task when the moving surface is operating at 10cm/s (a representative value for automotive assembly lines). For the same task, Fig. 6 shows the comparative performance of the nominal and modified system. In the case of nominal system, no information regarding the surface state is used resulting in significant deviation from the desired path and inaccurate localization. On the other hand, using the modified system the robot can successfully navigate across the assembly line dynamically compensating for the surface velocity and correcting its heading.

[5]`http://goo.gl/8JGtVQ`
[6]`https://www.sparkfun.com/products/9208`

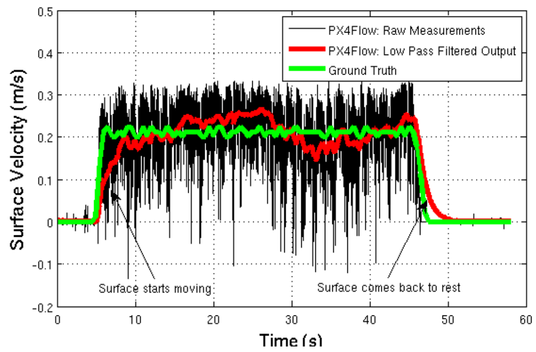

Fig. 4. Performance of the optic flow sensor as compared to the ground truth. Both raw measurements from the optic flow sensor, and its filtered value using low pass filter are shown.

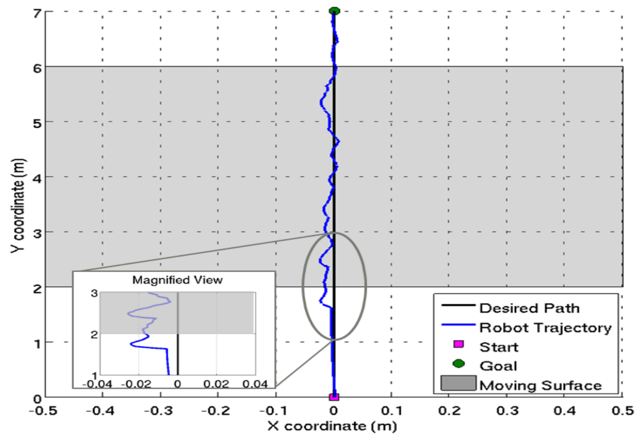

Fig. 5. Performance of the modified system for the task of robot traversing the moving surface from a typical Gazebo simulation run. Magnified view shows the deviation in the path as the robot enters the moving surface. However, this deviation is <4cm for the entire path, validating the modification in control sub-system.

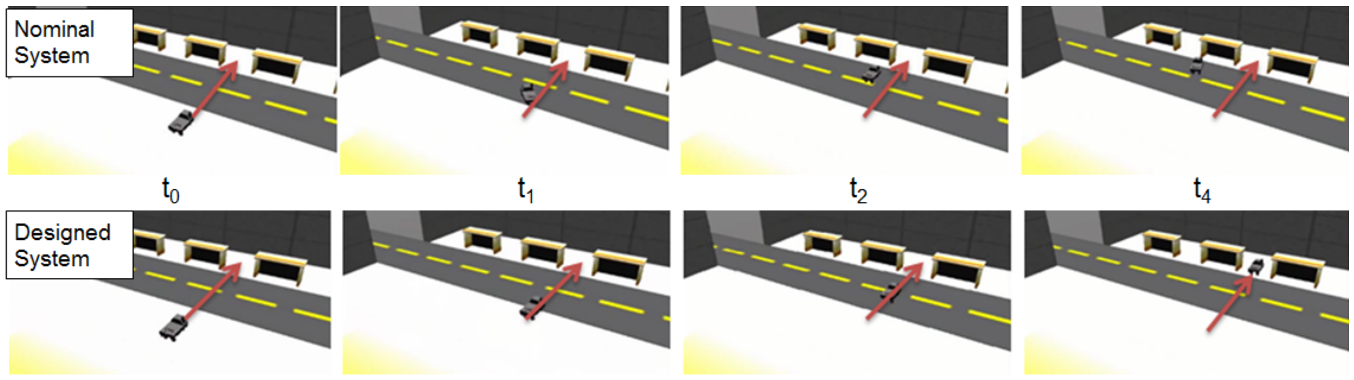

Fig. 6. Four stills from the simulated performance of the modified control architecture (below) compared with the nominal architecture (top), while the robot traverses the moving surface. The red arrow indicates the desired robot path. The robot tries to move across the moving surface, but can do so successfully only with the modified system (as seen in the bottom figures). For complete simulation run see the accompanying video.

C. Initial Hardware Validation

After independent validation of the controller (through Gazebo simulation) and optic flow sensor (via comparison with the ground truth), we proceed towards observing the performance of the combined system. As an initial step, the sensor for surface velocity is mounted off-board and performance of the system for the 'position hold' task is observed. Position hold is an important task for a robotic assistant, especially while delivering parts to a human associate working on a moving line.

Fig. 7 shows the two test scenarios for the position hold experiment, namely, when the robot is completely on, and when it straddles the moving surface. For these tests, the customized treadmill operates at ≈20cm/s during its on state. For both the cases, the robot successfully maintains a constant position despite the change in the assembly line's on/off state. In these tests, the laser scanner based localization is used to detect when the wheels are on the customized treadmill and the surface velocity is obtained using the ground truth sensor; for surfaces with sufficient image features off-board optic flow sensors also yield similar performance. Next, the robot's ability to navigate across the moving assembly line is tested in the same test environment. Similar to Gazebo simulation, the robot successfully traverses the moving surface (see accompanying video at `http://tiny.cc/aogo3w`).

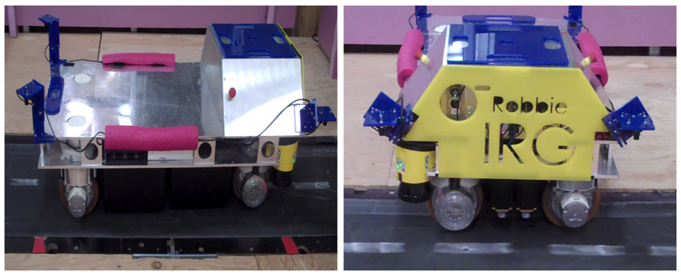

Fig. 7. Position hold experiments with off-board surface sensing being conducted in two cases: the robot being completely on (left), and straddling (right) the moving surface.

D. Limitations

In our problem description the assembly lines operate in two states (on/off), with the on state exhibiting uniform speed (<20 cm/s). Hence, our validation in simulation and hardware has been carried out for such a moving surface. The proposed solution in principle can be applied for moving surfaces with variable speed; however the performance will be limited by that of the optic flow sensors, which is dependent on the image features of the moving surface and measurement range. Further, for safety considerations it is desirable to introduce redundancy for the optic flow sensors. We are currently exploring additional measures to improve the reliability of surface sensing using position and velocity information obtained from laser scanner-based localization and encoder information.

VII. Conclusion and Future Work

Introducing a mobile robotic assistant to perform "non-value added work" in assembly lines can yield significant improvements in productivity. However, prior to accomplishing any task on the assembly line, the robot first needs to be able to traverse the moving floor present in assembly lines. In this paper we present a control and sensing solution for a mobile robotic assistant as it traverses a moving-floor assembly line. Information about surface velocity is essential for our modified control algorithm. Further, an alternate method is required for localization since encoder information degrades when the robot is navigating on moving surfaces. Off-the-shelf sensors and localization algorithms are explored for fulfilling this requirement, and a customized solution is presented using PX4Flow optic flow sensors and a laser scanner-based localization algorithm.

We perform initial validation of the controls and sensing subsystems using both software simulation and hardware tests. Software simulation and initial demonstrations of the full hardware system yield promising results; using off-board surface sensing the robot successfully moves across and maintains its position while on, and while straddling, the moving line. These initial demonstrations are currently being followed up with tests of the system with on-board sensing of surface motion.

The solution to the problem of traversing assembly lines is but a first step towards the deployment of mobile robotic assistants on the factory floor. To provide benefit, the robotic assistant requires path planning, deliberative task planning/scheduling, and human interface capabilities, ensuring delivery of the right materials to the point-of-use at the right time. Various challenges, in both research and implementation, need to be addressed.

In the current setup, we demonstrated operation of the robot on a moving surface but in an otherwise static, known environment. In a real factory setting, the environment will be dynamic with part carts and humans moving in a very dense and cluttered space. Additional capabilities are necessary to detect and avoid humans and other obstacles, and to ensure efficient robot navigation. Conventional path planning algorithms are reactive, relying primarily on current knowledge of obstacles in the map, to decide a future path for the robot. In dynamic environments, these algorithms often yield trajectories which are highly suboptimal and often require the robot to stop unnecessarily [27]. We anticipate the current application will require path planning approaches for dynamic environments that anticipate the trajectories and goals of obstacles [28]–[30]. Lastly, the robot will not only need to infer the intent of the human associates, but also have the ability to communicate its intent for a fruitful collaboration.

VIII. Acknowledgments

The authors thank Ron Wiken for help with the construction of customized treadmill for emulating the assembly line.

References

[1] R. Bostelman and W. Shackleford, "Improved performance of an automated guided vehicle by using a smart diagnostics tool," in *IEEE International Conference on Industrial Technology (ICIT)*, pp. 1688–1693, 2010.

[2] E. Guizzo, "Three engineers, hundreds of robots, one warehouse," *IEEE Spectrum*, vol. 45, no. 7, pp. 26–34, 2008.

[3] "Safety of human-robot systems in fixed workcell environments." `http://www.nist.gov/el/isd/ps/safhumrobsysfixedworkcellenvir.cfm`. [Online; last accessed 1-Sept-2013].

[4] "Baxter: manufacturing robot." `http://www.rethinkrobotics.com/products/baxter/`. [Online; last accessed 1-Sept-2013].

[5] "KUKA robots." `http://www.kuka-robotics.com/usa/en/pressevents/news/NN_051017_01_Teamworking.htm`. [Online; last accessed 1-Sept-2013].

[6] M. Treat, S. Amory, P. Downey, and D. Taliaferro, "Initial clinical experience with a partly autonomous robotic surgical instrument server," *Surgical Endoscopy And Other Interventional Techniques*, vol. 20, no. 8, pp. 1310–1314, 2006.

[7] "VECNA hospital delivery robots." `http://www.vecna.com/solutions/hospital-delivery`. [Online; last accessed 1-Sept-2013].

[8] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun, "Towards robotic assistants in nursing homes: Challenges and results," *Robotics and Autonomous Systems*, vol. 42, no. 3, pp. 271–281, 2003.

[9] I. R. Nourbakhsh, C. Kunz, and T. Willeke, "The mobot museum robot installations: A five year experiment," in *International Conference on Intelligent Robots and Systems (IROS)*, vol. 4, pp. 3636–3641, 2003.

[10] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot," *Artificial Intelligence*, vol. 114, no. 12, pp. 3 – 55, 1999.

[11] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, and N. Hagita, "A communication robot in a shopping mall," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 897–913, 2010.

[12] B. Graf, U. Reiser, M. Hagele, K. Mauz, and P. Klein, "Robotic home assistant Care-O-bot® 3-product vision and innovation platform," in *IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pp. 139–144, 2009.

[13] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. V. Weghe, "HERB: a home exploring robotic butler," *Autonomous Robots*, vol. 28, no. 1, pp. 5–20, 2010.

[14] J. C. Boerkoel Jr and J. A. Shah, "Planning for flexible human-robot co-navigation in dynamic manufacturing environments," in *International Conference on Human-Robot Interaction (HRI), Proceedings of Human Robot Interaction Pioneers*, 2013.

[15] V. V. Unhelkar, H. C. Siu, and J. A. Shah, "Comparative performance of human and mobile robotic assistants in collaborative fetch-and-deliver tasks," in *International Conference on Human-Robot Interaction (HRI)*, 2014.

[16] "Rob@Work 3-6." `http://www.care-o-bot-research.org/robatwork-3`. [Online; last accessed 1-June-2013].

[17] "Robot Operating System." `http://wiki.ros.org`. [Online; last accessed 1-June-2013].

[18] "Gazebo Simulator." `http://gazebosim.org/`. [Online; last accessed 1-June-2013].

[19] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *National Conference on Artificial Intelligence (AAAI)*, vol. 1999, pp. 343–349, 1999.

[20] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[21] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[22] 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 *International Conference on Robotics and Automation (ICRA)*, 2013.

[23] 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 *47th IEEE Conference on Decision and Control, CDC 2008*, pp. 4976–4983, Dec 2008.

[24] "InnoSenT miniature radars." `http://www.innosent.de/en/industry/isys-systems`. [Online; last accessed 1-June-2013].

[25] I. Baldwin and P. Newman, "Laser-only road-vehicle localization with dual 2d push-broom lidars and 3d priors," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 2490–2497, 2012.

[26] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable SLAM system with full 3d motion estimation," 2011.

[27] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 797–803, 2010.

[28] S. Thompson, T. Horiuchi, and S. Kagami, "A probabilistic model of human motion and navigation intent for mobile robot path planning," in *International Conference on Autonomous Robots and Agents (ICARA)*, pp. 663–668, 2009.

[29] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 3931–3936, 2009.

[30] P. Trautman, "Probabilistic tools for human-robot cooperation," in *International Conference on Human-Robot Interaction (HRI), Human Agent Robot Teamwork Workshop*, 2012.