An Integrated Robotic System for Autonomous Brake Bleeding in Rail Yards

Huan Tan, Shiraj Sen, Arpit Jain, Shuai Li[1],Viktor Holovashchenko, Ghulam Baloch, Omar Al Assad, Romano Patrick, Douglas Forman, Yonatan Gefen, Pramod Sharma, Frederick Wheeler, Charles Theurer, Balajee Kannan

balajee.kannan@ge.com[2]
GE Global Research
One Research Circle
Niskayuna, USA

Abstract. Current operations in rail yards are dangerous and limited by the operational capabilities of humans being able to make critical decisions in the presence of incomplete or incorrect information. Such issues call out the need for robust and capable autonomous systems. In this paper, we outline one such autonomous solution for the railroad domain, capable of performing the brake bleeding inspection task in a hump yard. Towards that, we integrate a large form factor mobile robot (the Clearpath Grizzly) with an industrial manipulator arm (Yasakawa Motoman SIA20F) to effectively detect, identify and subsequently manipulate the brake lever under harsh outdoor environments. In this paper, we focus on the system design and the core algorithms necessary for reliable and repeatable system execution. To test our developed solution, we performed extensive field tests in a fully operational rail yard with randomly picked rail cars under day and night-time conditions. The results from the testing are promising and validate the feasibility of deploying an autonomous brake bleeding solution for railyards.

Keywords: Field Robotics, Autonomous System, 3D Detection & Segmentation, Manipulation, Trajectory Planning, System Design, Rail Yard Operations, Brake Bleeding

1 Introduction

The challenges in the modern rail yard are vast and diverse. Railroad classification yards, or hump yards, play an important role as consolidation nodes in rail freight networks. At classification yards, inbound trains are disassembled and the railcars are sorted by next common destination (or block) [1]. Based on various studies for the rail industry, the efficiency of yards in part drives the efficiency of the entire rail network [2]. The hump yard is generally divided into three main areas: the receiving

[1] Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180
[2] Communicating author

adfa, p. 1, 2011.

yard, where inbound trains arrive and are prepared for sorting, the class yard, where railcars are sorted into blocks and the departure yard, where blocks are assembled into outbound trains, inspected and then depart.

In the future, rail yards will be a hub of technological advancements where autonomous mobile robotic solutions work in coordination with human yard operators and cloud-based data analytics to enhance human safety [3] and drive services productivity (see Fig.1).

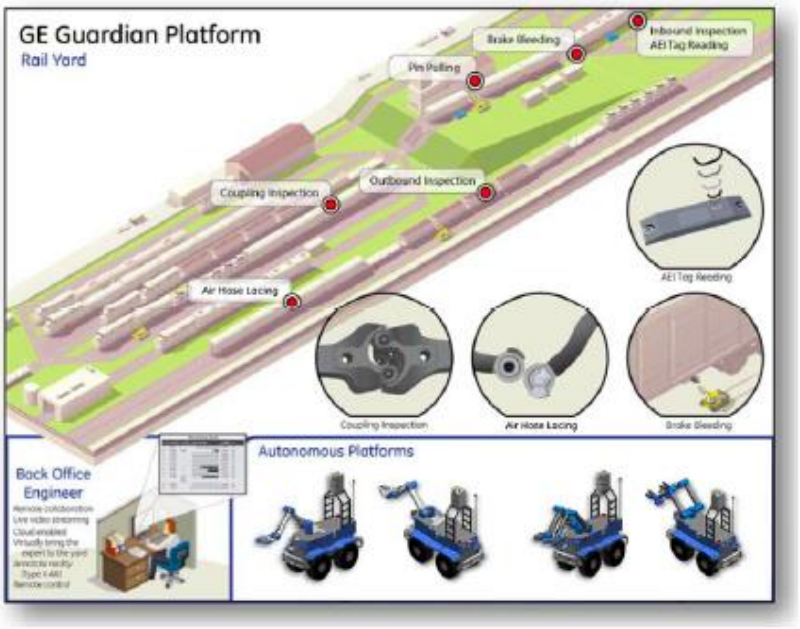

Fig. 1. GE Guardian Platform for Autonomous Railyard Inspection Services

Current solutions for field service operations are labor-intensive, dangerous, and limited by the operational capabilities of humans being able to make critical decisions in the presence of incomplete or incorrect information. Furthermore, efficient system level-operations require integrated system wide solutions, more than just point solutions to key challenges [4]. The nature of these missions dictates that the tasks and environments cannot always be fully anticipated or specified at the design time, yet an autonomous solution must have the essential capabilities and tools to carry out the mission even if it encounters situations that were not expected.

Solutions for typical railyard problems, such as brake bleeding, brake line lacing, coupling cars, etc, require combining mobility, perception, and manipulation towards a tightly integrated autonomous solution. Currently, there exist a number of mobile manipulation systems that have been successfully demonstrated in controlled indoor environments. An exhaustive survey of such systems and techniques is beyond the scope of this paper. Ciocarlie et. al [5] showed how a PR2 robot can utilize a 3D sensor for object detection and manipulation in cluttered tabletop environments. The Intel

HERB mobile manipulation platform has demonstrated impressive capabilities for picking up and placing of objects in indoor environments [6]. Nguyen et. al [7] presented the El-E robot that could autonomously fetch objects from flat surfaces. All these platforms make certain assumption with regards to the environment in order to simplify the problem of robot perception. When placing robots in an outdoor environment, technical challenges increase largely, but field robotic application benefits both technically and economically [8]. Some researchers have used robots in the domains of agriculture [9] [10], mining [11] [12], transporting [13] [14], etc. The DARPA Robotics Challenge (DRC), aimed at advancing the capabilities of robotic systems for disaster response, is one of the few examples of robots working in unstructured outdoor environments [15]. This challenge required a robot to collaborate with humans in order to achieve varied task objectives such as opening a door, turning a valve, and drilling a hole. The difficulty associated with detecting objects reliably in outdoor environment results in the problem of object detection primarily being performed by a human operator, fitting models to sensory data [16].

One key challenge in yard operation is that of bleeding brakes on inbound cars in the receiving yard. Railcars have pneumatic breaking systems that work on the concept of a pressure differential. Fig.2 displays different types of brake levers robots need to grasp and pull. Interestingly, the size of the brake lever is significantly small compared to the size of the environment and the rail cars. In addition, there are lots of variations on the shape, location, and the material of the brake levers. Coupled with that is the inherent uncertainty in the environment; every day, rail cars are placed at different locations, and the spaces between cars are very narrow and unstructured.

In our vision, we believe that intelligent autonomy, robust perception, and robust actuation are the keys to a successful autonomous robotic system. In this paper we outline one such autonomous solution for performing the brake bleeding task in a rail yard environment. We believe our solution is one of the first of its kind in automating the brake bleeding process for rail yards, addressing a real need for robotics in the domain of industrial applications.

The rest of the paper is organized as follows: Section 2 explains the system design in detail, Section 3 discusses our experimental results, and Section 4 summarizes the results and identifies future work scope.

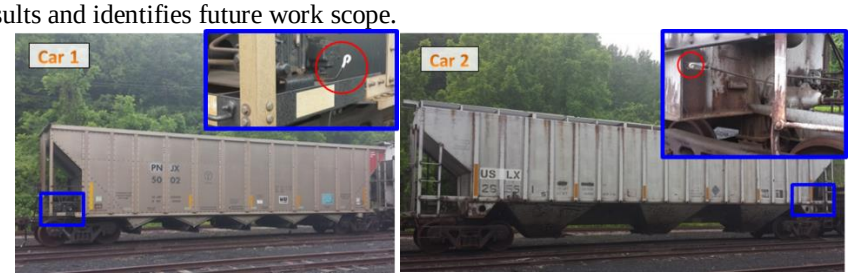

Fig. 2. Typical Cars and Brake Levers

2 System Design

The developed system is used to actuate brake levers on rail yards, which is a typical field robotic operation. The system involves the robot autonomously navigating within the track corridor along the length of the train moving from car to car (given an initial coarse estimate of the brake rod location from rail database), locating the brake rod, positioning itself next to the brake rod before actuating the brake rod. During the autonomous navigation, the robot needs to maintain a distance of separation (0-4") from the plane of the railcar while moving forward. In order to ensure real-time brake rod detection and subsequent estimation of the brake rod location, a two-stage detection strategy is utilized. Once the robot has navigated near to the brake rod location an extremely fast 2D vision-based search algorithm confirms a coarse location of the brake rod. The second stage of the algorithm involves building a dense model for template-based shape matching (brake rod) to identify the exact location and pose of the break rod. Once the robot reaches the desired location, the final step in the process is to actuate and manipulate the rod with a robotic arm. The manipulation solution needs to be flexible enough to handle variations in the environment, the location and state of the rod and in sensing and perception. The detail of the grasping strategy is beyond the scope of this paper.

2.1 System Architecture

The brake bleeding architecture is composed of three layers: Physical Layer, Processing Layer, and Planning Layer (see Fig. 3). The physical layer deals with the robots and sensors. For the physical layer, we leverage commercially available sensors and platforms integrating to build a cohesive and highly capable platform. The details of the actual platform are described later in Section 3.

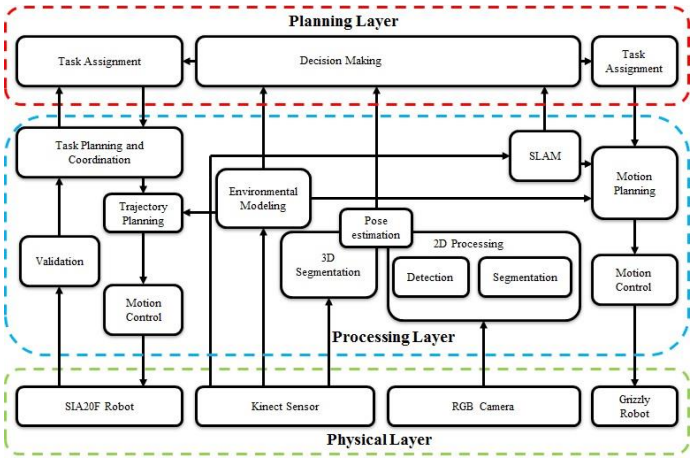

Fig. 3. System Architecture

The processing layer is the crucial part of the system, since most of the algorithms and functions are implemented on this layer. Based on the fields of study, we categorize the major components into four modules: *Deliberation, Perception, Navigation, and Manipulation*. Deliberation module is responsible for planning and coordinating all the behaviors in the system. It gets processed information from sensors and makes decisions to move the Grizzly robot and the SIA20F robot. Perception module processes sensory information and find the poses of brake levers in the environment. Navigation module controls the Grizzly robot to move to desired position. In order to move safely and precisely, a variant of RTAB-Map [17] algorithm based on Environmental Modeling is implemented to provide necessary information for motion planning. Manipulation module is used for controlling the SIA20F robot to touch the lever. Environmental Modeling is used to help the robot to plan the motion trajectories while avoiding collision.

Finally, in the planning layer, all sensory information and states information are collected from lower layers towards action selection. According to requirements of a task, the system will make different decisions based on current task-relevant situation. A state machine ties all the layers together and transfer signals from navigation to perception and then subsequently manipulation. Whenever there is an emergency stop signal generated or there are error information reported by three sub-modules, the system safety primitives are triggered, preventing any damages to both the robot and the environment.

2.2 Environmental Modeling and Platform Motion Control

Moving the platform between different locations in rail yards to perform tasks is a basic requirement for mobile manipulation systems. We provide two operation modes in the navigation module: one is the teleoperation mode, and the other is autonomous mode. Using the triggered signal from a joy stick, the deliberation module sends a command to the navigation module to switch the mode. In the autonomous mode, the robot moves to search brake levers in the environment. The navigation module is tightly coupled with the detection module, moving the robot in a trajectory parallel to the plane of the railcar until a lever is found. Fig.4 (next page) displays the state machine for the navigation module.

Environmental Modeling.

In order to move in the environment, the robot needs to model and understand the environment. Currently, the robot motion is limited to being able to navigate the length of a train (typically about 100 rail cars long) and subsequently does not need to move long distance, which means global planning is not required. However, even in the local planning and movement, the robot needs to safely operate in its environment. Towards that, we implemented a Kinect-based SLAM algorithm in our system. There are lots of SLAM algorithms off-the-shelf and we choose Real-Time Appearance-Based Mapping (RTAB-Map) [17], which is a RGB-D Graph-Based SLAM approach

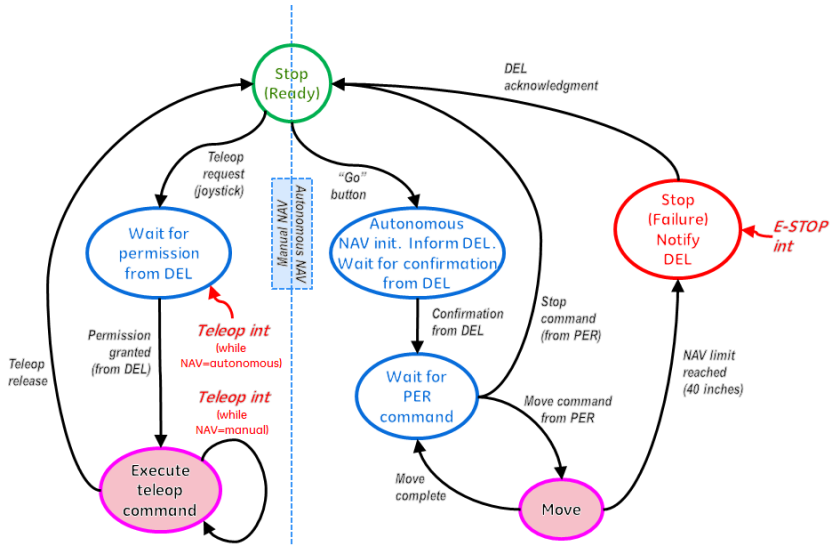

Fig. 4. Navigation State Machine

based on an incremental appearance-based loop closure detector. In our testing, we found it to be very stable under most of the weather conditions. Using RTAB-Map, the robot knows the relative location of itself, which is used to close the motion control loop. At the same time, we use point cloud data to recognize the plane of the cars. This information is used to keep the robot away from the rail cars and maintain a predefined distance of separation (see Fig.5, next page).

Motion Planning and Control.

There are two planning modules implemented in our system. One is to move minimize the distance between the robot and the desired location, and the other is to keep safe distance between the robot and the tracks. The computed control commands from the goal controller for driving the robot forward and from the safety controller for keeping the robot away from rail cars are fused to compute a velocity command which will control the motors of the robot to move in the environment. The fusion is a weighted sum of the two control commands:

$$cmd_{vel} = \alpha * cmd_{goal} + \beta * cmd_{safety} \quad (1)$$

$$\alpha + \beta = 1 \quad (2)$$

cmd_{goal} and cmd_{safety} are generated using the Artificial Potential Field [18] algorithm. α and β are parameters tuned based on the task-relevant situations.

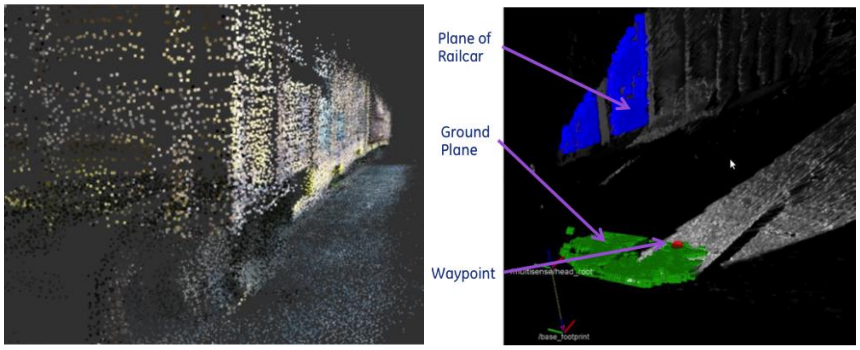

Fig. 5. Point Cloud Building and Subsequent Waypoint generation for Navigation

2.3 Target-of-Interest Detection and Pose-Estimation [19]

Detecting and finding the correct 6D pose of the correct brake lever is a necessary first step to actuating the brake lever. Our approach for object detection in unconstrained environments relies on obtaining object hypothesis candidates by fusing detection results from 2D images and 3D point clouds. The candidates are then combined temporally and reasoned upon in an online fashion by using real time Simultaneous Localization and Mapping (SLAM). We propose a confidence function that ranks the candidate detections based on its detection score, spatial, and temporal consistency. The confidence function also takes into account the uncertainty in detection location due to occlusion or SLAM misalignment. Fig.6 shows the pipeline of our approach.

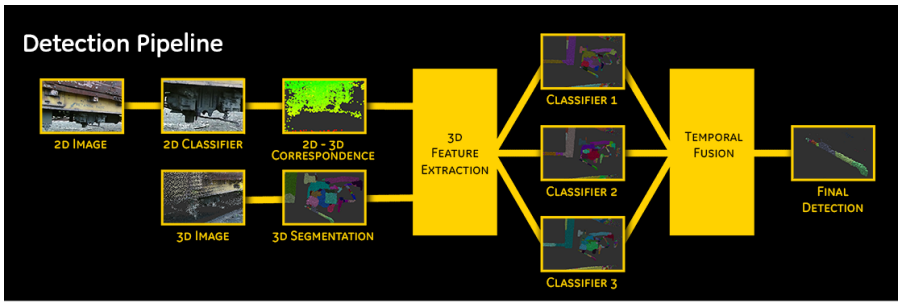

Fig. 6. Detection and Pose Estimation

Image based Object Detector

We use deep learning to train an object detector from images. Our detector utilizes the layered architecture of deep CNN [20] to learn the object representation. Each layer comprises of linear and non-linear operators that are learned jointly in an end-to-end system using a task specific loss function. The goal of CNN is to learn a feed-forward neural network defined as:

$$f(x) = f_L(\dots f_2(f_1(x; w_1); w_2) \dots ; w_L) \quad (3)$$

Each f_i represents a layer that takes data x_l from previous layer and a parameter w_i and produces an output x_{l+1} for next layer. The parameters of a CNN $w = (w_1, w_2, \dots, w_L)$ are learned using back-propagation to reduce the overall classification error [21].

We bootstrap the training images with the ImageNet model [22] trained on the AlexNet architecture. Deep learning detectors have high precision provided there is enough training data. However their runtime performance is slow due to the high computational requirements of the multilayered neural network. In this paper, a Deformable Part-based Model (DPM) [23] detector is used to filter the image and reduce the number of object candidates being passed to the neural network. A DPM learns the appearance and spatial arrangement of the object and its parts using histogram of oriented gradient features. The top N candidates above a threshold that are returned by the deep learning classifier are selected as potential object hypotheses. An Ensemble of Shape Functions (ESF) [24] is extracted from each of the 3D segments. Three histograms are generated from each of these shape functions, including the histogram of the connecting lines generated from random points that lie on the object surface, the histogram of connecting lines that do not lie on the object surface, and the histogram for the case when part of the connecting lines lie on the object surface. The 3D feature descriptor comprising of multiple histograms is encoded into a single high dimensional feature descriptor by using Fisher Vector encoding. Two sets of classifiers were trained using Support Vector Machines (SVM) [25] and Real Ad-Boost. Multiple classifiers allow us to leverage strengths of individual classifier to obtain a robust score for detection.

3D Segmentation

Object proposals from 3D point cloud are obtained through segmentation process. We use Locally Convex Connected Patches (LCCP) [26] algorithm to segment the 3D point cloud [27]. LCCP is a bottom up approach that merges super voxels into object parts based on a local convexity/concavity criterion. A basic filtering step based on size and shape constraints are applied to these segments to reduce the object hypothesis space.

Temporal Fusion

A voting scheme is utilized for hypothesis evaluation, wherein each hypothesis votes for the position of the object.

The confidence of the object being at a particular location is computed by

$$Confidence(l_j) = \mu_j * e^{(\mu_j * Count_j - \delta_j)} \quad (4)$$

where μ_j and δ_j are the means and variances of classifier scores for all segments voting for location l_j , and $Count_j$ is the number of segments voting for that particular location.

2.4 Trajectory planning and actuation

The ultimate goal is to actuate (touch) brake levers on rail cars. After the detection is done, the robot knows where the target of interest, i.e., the brake lever, is. Using the pose estimation results, the robot plans a motion trajectory in the workspace based on the current environmental modeling results. Then the controller drives the arm safely to touch the brake lever. Fig.7 displays the state machine of manipulation.

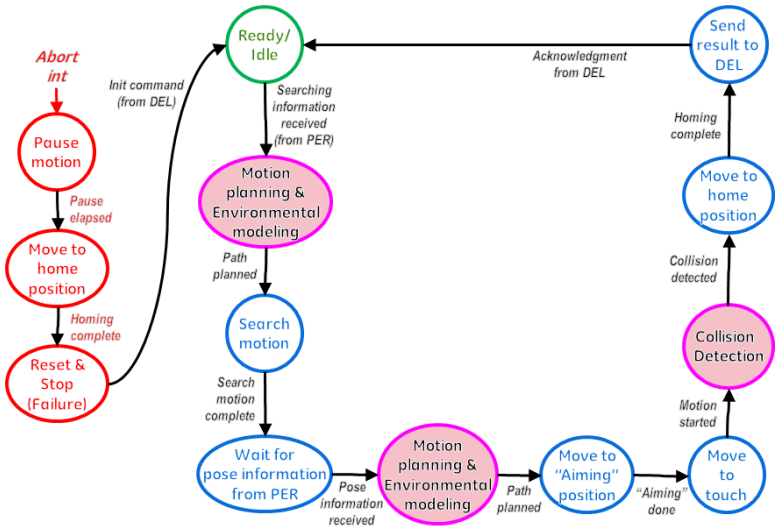

Fig. 7. Manipulation State Machine

Environmental Modeling

Our system is used to touch brake levers on rail cars, which means the majority of the motions are "moving toward". In our testing, we found that grid-based algorithm is enough for modeling our environment.

The environmental data is collected using Kinect. All the point cloud data points are processed and grouped into grid, i.e., OctoMap [28]. Fig.8 (next page) displays an Example of modeling the environment. The grid are cubes with the sizes of 10cm*10cm*10cm. The Kinect can detect the point cloud data points up to 8m. In order to speed up the modeling and analysis process, we only model the environment around the manipulator, which is in a sphere with the radius 2.5m.

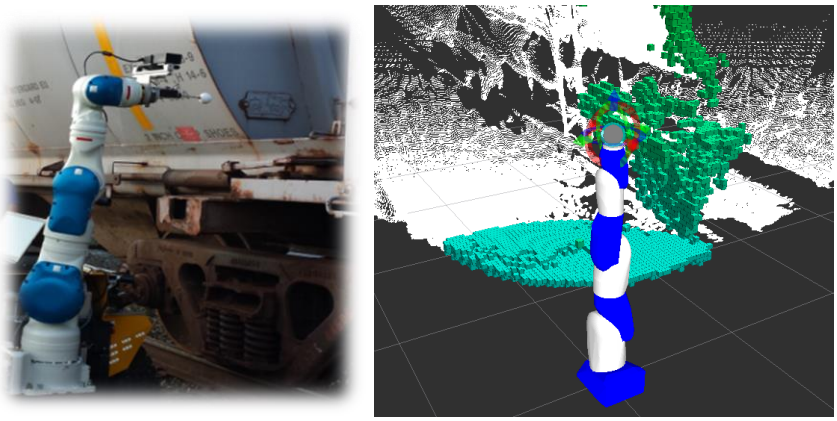

Fig. 8. Environmental Modeling

Trajectory Generation

Two requirements are set up for the motion planning algorithm: 1. Fast; 2. Safe. We used the Robot Operating System (ROS) [29] Motoman Package [30] developed by ROS-Industrial Consortium as the basic motion planning driver and created three ROS services, including linear trajectory planning in joint space, linear trajectory planning in Cartesian space, and Point-to-Point trajectory planning in joint space, for motion planning in our system.

Linear Trajectory Planning in Joint Space.

When the robot is moving in an open space for a long distance and far away from the cars and brake levers, we do not need to worry about the motion patterns. Then we define the starting position and target position of the motion. Using Artificial Potential Field algorithm, we can find the way points on the desired motion, which is linear in all 6 degrees of freedom, but non-linear in the Cartesian space. Then we assign velocities to each way points depending on the task requirements.

When integrated with the deliberation module, we cannot use this method directly. Because the positions of the brake levers are defined as 6D poses in Cartesian space. After get the 6D poses in the Cartesian space, using Inverse Kinematics, we convert the 6D pose from Cartesian space to 6 joint angles in the joint space. Then this service can compute the desired joint angles on the motion trajectory.

Linear Trajectory Planning in Cartesian Space.

This service is an enhanced version of the first service. We use Artificial Potential Field algorithm to find the way points on a desired motion trajectory in the Cartesian space. Using Inverse Kinematics, we get corresponding way points in the joint space. Then we assign velocities to all the way points.

Point-to-Point Trajectory Planning in Joint Space.

This service is created for homing or resetting. We just defined the target joint angles without any internal way points. This method could bring the robot to the target position as fast as possible.

Motion Control

The generated motion trajectories are defined as a sequence of way points in the joint space. Each way point contains the information of 7 joint angles, timing stamp, and velocity. The joint angles, timing stamp, and velocity are put in the vector of points. The standard MotoROS program is run on the robot to receive the command to drive the robot to move along the desired motion trajectory.

Validation

A microswitch produced by Honeywell is attached to the wrist of the SIA20F robot. And the other end is a white ball. Whenever the ball touches the brake lever, the microswitch will be triggered to provide feedback information to the robot.

3 Experimental Results and Discussion

The proposed design was developed and validated on a practical mobile robotic platform. Fig.9 displays the created system. We use a Clearpath Grizzly mobile robot as our base platform integrated with a 7-degree of freedom Yasakawa Motoman SIA20F manipulator arm coupled with a custom end effector. Towards environmental modeling and brake lever detection, we used two RGB cameras for 2D processing including detection and segmentation in conjunction with two Kinect sensors. The point cloud information is also used for mapping and localization for the mobile platform.

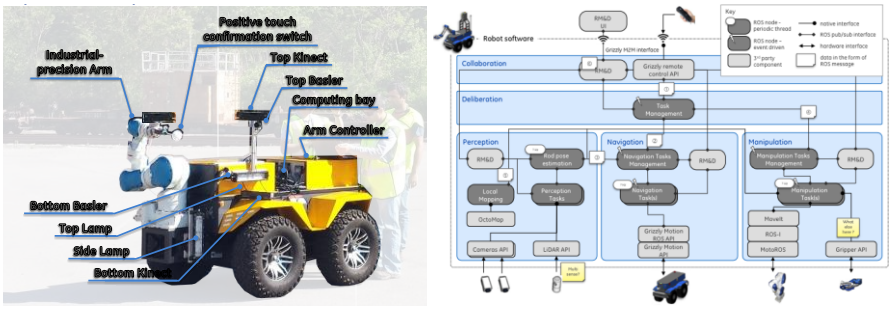

Fig. 9. Development Platform (hardware and software centric views)

Two Lenovo W541 laptops, which are equipped with Intel Core i7-4910MQ 2.90GHZ GPU and 32.0G RAM Memory, are used in our system. One is responsible for motion control of the mobile platform, detection and pose-estimation of targets, and high level state machine. The other is majorly used for controlling the SIA20F robot to plan the motion trajectories and physically touch brake levers. Robot Operating System (ROS) is used as the low-level IPC for our system.

3.1 Detection Results

The performances of the image-based detectors were evaluated on 2000 color images collected from 80 test rail cars. Fig.10 shows the performance of the two image-based detectors. As is evident, the CNN detector performs better than the DPM. This is because the deep learning architecture doesn't reply on hand designed features but learns the appropriate features directly from the data that are more discriminative. There is very little loss in performance when the CNN detector takes as input the candidate patches selected by DPM. However, using the detectors in sequence leads to significant improvement in detection speed compared to the CNN detector running on the entire image.

The performance of the 3D feature descriptors were evaluated by randomly dividing the positive and negative 3D segments into training and testing data and training a Real AdaBoost classifier. We evaluated the performance of our system for braking mechanism detection and touching on 80 test rail cars.

In our approach, with SLAM incorporated, our system achieved 85% detection performance. This demonstrates that temporally fusing detection results significantly improve object detection accuracy. Moreover, our perception system performed equally well on detection task during the day and the night time validating that our approach is robust to environmental and illumination changes.

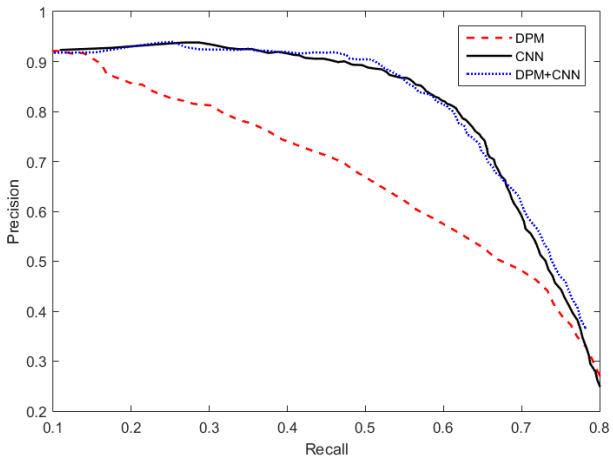

Fig. 10. Figure shows the precision-recall curve for the Deformable part-based model detector and the detector trained using deep learning. The performance of the combination of DPM and CNN detector is comparable to CNN-based detector.

3.2 Motion Planning Results

After we successfully find a brake lever, we need to plan a trajectory for the SIA20F to touch the lever. When the environment is open and there are no obstacles in the

environment, as long as the brake lever is in the reachable space of the manipulator, the trajectory planning is 100% successful.

3.3 System Testing Results

In GE's Railcar repair facility in Sayre, PA (see Fig.11), we evaluated our system on 61 rail cars over multiple days and times (see Fig.11) to account for varying environment conditions. For our tests, we randomly picked cars in the rail with the distribution of the cars ranging from hoppers to tankers, box and gondolas. For testing, we restrict the search distance to about 2.5m. A test is considered as a success if the robot touches the designated brake lever, otherwise it is a failure. We randomly picked rail cars from our rail yard and tested 61 times.

Fig. 11. Test Environment (GE Railcar Repair Facility) and test conditions (day & night)

(a) Test railcar set (b) Robot Navigation (c) Brake Lever Detection

(d) Trajectory Generation (e) Motion Execution (f) Successful Actuation

Fig. 12. Autonomous Brake Bleeding Operations – from left to right; the robot is lined next to a set of railcars, navigates to the first one, detects the brake lever, moves into position adjacent to the brake lever, calculates appropriate trajectory and actuates brake lever.

Fig. 12 displays the scenario of the testing process: (a) the robot is placed in a rail yard, where it needs to handle the brake levers along a rail track, (b) the robot is ini-

tially placed on a location about 2.5m away from the brake lever and it moves along a rail track, (c) the robot keeps detecting the brake lever in the environment using visual information, (d) the robot stops at a location and plans a motion trajectory to move the robot manipulator to touch the brake lever, (e) the arm moves toward the brake lever, and (f) touches the brake lever without hitting obstacles in the environment. After the lever is touched, the robot moves to the next location.

Raw Testing Results

In 61 trials, our system successfully touched the brake levers 41 times. The overall success rate of touching is 67.21%. Table 1 displays the original data collected.

Table 1. Original Testing Results

Error Type	Occurrence
1 - Perception - Not found	1
2 - Perception - Wrong detection	7
3 - Perception - Back to prior detection	0
4 - Perception - Undefined	1
5 - Navigation	1
6 – Navigation - Motor Malfunction	4
7 - Human error - Positioning	3
8 - Human error - Undefined	1
9 - Manipulation - Obstacle Collision	0
10 - Manipulation - Motion Incomplete	1
11 - Other	1
	Total errors: 20
	Overall Success Rate: 67.21%

Adjusted Testing Results.

In the 20 failed trials, we categorized the cause of failure into four types: Detection Error, Navigation Error, Manipulation Error, and Human Operation Error. The original distribution of the causes of errors is displayed in Table 2.

Table 2. Error Distribution

Trials	Categorization	Percentage
Success	Successful	67.21%
Failure	Detection Error	14.75%
	Navigation Error	8.20%
	Manipulation Error	1.64%
	Human Operation Error	8.20%

Detection errors and Manipulation errors are produced due to technical reasons. For example, the detection module may find a wrong lever, which is a positive false

error; the detection module may not be able to find any levers in the environment, which is a negative true error. However, the human operation error can be removed after the users are well trained. 4 navigation errors are produced by the malfunction of the motors. If we replace those motors, the errors can also be removed. Then we remove the human operation errors and 4 navigation errors, the adjusted result is displayed in Table 3.

Table 3. Adjusted Testing Results

Trials	Categorization	Percentage
Success	Successful	81.97%
Failure	Detection Error	14.75%
	Navigation Error	1.64%
	Manipulation Error	1.64%
	Human Operation Error	0.0%

4 Conclusion

This paper proposes an integrated system design of deploying mobile robots in rail yards for autonomous operations. We integrate two robots to perform autonomous brake bleeding in rail yards and the system design is explained in detail in this paper. The developed system has been tested in a rail yard and the experimental results validate that our system can successfully navigate, find, and touch brake levers. As mentioned earlier, this is an on-going project. The next step is to develop grasping and pulling capabilities for the industrial manipulator. The system will be tested more in more rail yards under different conditions.

References

1. https://en.wikipedia.org/wiki/Rail_yard
2. Oum, Tae Hoon, W. G. Waters, and Chunyan Yum (1999) A survey of productivity and efficiency measurement in rail transport. Journal of Transport economics and Policy: 9-42.
3. Drudi, Dino. "Railroad-related work injury fatalities." Monthly Lab. Rev. 130 (2007): 17.
4. Thorpe, Chuck, Hugh Durrant-Whyte (2001). Field robots. Proceedings of the 10th International Symposium of Robotics Research (ISRR'01)
5. Meeussen, Wim, Melonee Wise, Stuart Glaser, Sachin Chitta, Conor McGann, Patrick Mihelich, Eitan Marder-Eppstein (2010) Autonomous door opening and plugging in with a personal robot. Proceedings of 2010 IEEE International Conference on Robotics and Automation (ICRA): 729-736
6. Srinivasa, Siddhartha S., Dave Ferguson, Casey J. Helfrich, Dmitry Berenson, Alvaro Collet, Rosen Diankov, Garratt Gallagher, Geoffrey Hollinger, James Kuffner, and Michael Vande Weghe (2010) HERB: a home exploring robotic butler. Autonomous Robots 28(1): 5-20.

7. Nguyen, Hai, Cressel Anderson, Alexander Trevor, Advait Jain, Zhe Xu, and Charles C. Kemp (2008) El-e: An assistive robot that fetches objects from flat surfaces. Proceedings of the 2008 International Conference on Human-Robot Interaction
8. Pedersen, Søren Marcus, Spyros Fountas, Henrik Have, and B. S. Blackmore (2002) Agricultural robots: an economic feasibility study. Precision Agriculture 5: 589-595
9. Blackmore, B. S. (2007) A systems view of agricultural robots. Proceedings 6th European conference on precision agriculture (ECPA): 23-31
10. Redhead, Fiona, Stephen Snow, Dhaval Vyas, Owen Bawden, Ray Russell, Tristan Perez, and Margot Brereton (2015) Bringing the Farmer Perspective to Agricultural Robots. Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems:1067-1072
11. Green, Jeremy (2012) Underground mining robot: A CSIR project. Proceedings of 2012 IEEE International Symposium on Safety, Security, and Rescue Robotics: 1-6
12. Skonieczny, Krzysztof, Matthew Delaney, David S. Wettergreen, and William L. "Red" Whittaker (2013) Productive Lightweight Robotic Excavation for the Moon and Mars." Journal of Aerospace Engineering 27(4)
13. Jensen, Martin Andreas Falk, Dionysis Bochtis, Claus Grøn Sørensen, Morten Rufus Blas, and Kasper Lundberg Lykkegaard (2012) In-field and inter-field path planning for agricultural transport units. Computers & Industrial Engineering 63(4): 1054-1061
14. Suessemilch, Irene, Christoph Rohrer, Ramona Roesch, Clemens Guenther, Yorck Von Collani, Stephanie Linder, and Volker Fischer (2014) Projection Unit for a Self-Directing Mobile Platform, Transport Robot and Method for Operating a Self-Directing Mobile Platform." U.S. Patent Application 14/447,501, filed July 30, 2014.
15. Pratt, G., & Manzo, J. (2013). The DARPA robotics challenge [competitions]. IEEE Robotics & Automation Magazine 20(2): 10-12.
16. Johnson, Matthew, Brandon Shrewsbury, Sylvain Bertrand, Tingfan Wu, Daniel Duran, Marshall Floyd, Peter Abeles (2015) Team IHMC's Lessons Learned from the DARPA Robotics Challenge Trials. Journal of Field Robotics 32 (2): 192-208.
17. Labbe, Mathieu, and François Michaud (2014) Online global loop closure detection for large-scale multi-session graph-based slam. Proceedings of 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems: 2661-2666.
18. Khatib, Oussama (1986) Real-time obstacle avoidance for manipulators and mobile robots. The international journal of robotics research 5(1): 90-98.
19. Li, Shuai, Arpit Jain, Pramod Sharma, Shiraj Sen (2016), Robust Object Detection in Industrial En-vironments by using a Mobile Robot
20. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012) Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems: 1097-1105
21. LeCun, Yann, and Yoshua Bengio (1995) Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks 10.
22. Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009) Imagenet: A large-scale hierarchical image database. Proceedings of 2009 IEEE Conference on Computer Vision and Pattern Recognition: 248-255
23. Felzenszwalb, Pedro, David McAllester, and Deva Ramanan (2008) A discriminatively trained, multiscale, deformable part model. Proceedings of 2008 IEEE Conference on Computer Vision and Pattern Recognition : 1-8
24. Wohlkinger, Walter, and Markus Vincze (2011) Ensemble of shape functions for 3d object classification. Proceedings of 2011 IEEE International Conference on Robotics and Biomimetics: 2987-2992

25. Suykens, Johan AK, and Joos Vandewalle (1999) Least squares support vector machine classifiers. Neural processing letters 9(3): 293-300
26. S. Stein, F. Worgotter, M. Schoeler, J. Papon, and T. Kulvicius (2014) Convexity based object partitioning for robot applications. Proceedings of 2014 IEEE International Conference on Robotics and Automation: 3213–3220.
27. Rusu, Radu Bogdan, and Steve Cousins (2011) 3d is here: Point cloud library (PCL). Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA): 1-4
28. Wurm, Kai M., Armin Hornung, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard (2010) OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. Proceedings of the 2010 ICRA workshop on best practice in 3D perception and modeling for mobile manipulation (2)
29. Quigley, Morgan, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng (2009) ROS: an open-source Robot Operating System. Proceedings of 2009 ICRA workshop on open source software 3(2): 5
30. http://wiki.ros.org/motoman_driver