Towards Functional Labeling of Utility Vehicle Point Clouds for Humanoid Driving

Christopher Rasmussen

Dept. Computer & Information Sciences
University of Delaware
Newark, DE 19716
Email: cer@cis.udel.edu

Karthikeyan Yuvraj, Richard Vallett, Kiwon Sohn, Paul Oh

Dept. Mechanical Engineering, Drexel University
Philadelphia, PA 19104
E-mail: karthikeyan.yuvraj@gmail.com, rjvallett@gmail.com,
skw1125@gmail.com, paul@coe.drexel.edu

***Abstract*—We present preliminary work on analyzing 3-D point clouds of a small utility vehicle for purposes of humanoid robot driving. The scope of this work is limited to a subset of *ingress*-related tasks including stepping up into the vehicle and grasping the steering wheel. First, we describe how partial point clouds are acquired from different perspectives using sensors including a stereo camera and a tilting laser range-finder. For finer detail and a larger model than one sensor view alone can capture, a Kinect Fusion [1]-like algorithm is used to integrate the stereo point clouds as the sensor head is moved around the vehicle. Second, we discuss how individual sensor views can be registered to the overall vehicle model to provide context, and present methods to estimate several geometric parameters critical to motion planning: (1) the floor height and boundaries defined by the seat and the dashboard, and (2) the steering wheel pose and dimensions. Results are compared using the different sensors, and the usefulness of the estimated quantities for motion planning is also demonstrated.**

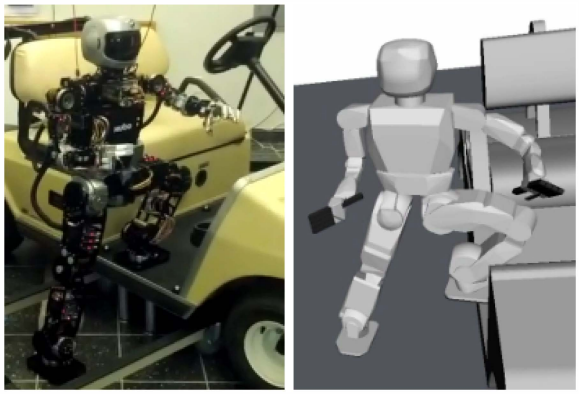

Fig. 1. (Left) Hubo robot stepping onto test utility vehicle; (right) Simulation of stepping motion

I. Introduction

As part of the recently-commenced DARPA Robotics Challenge (DRC) [2], contestant robots are challenged to approach a car-like vehicle and enter it, drive it to a target location in a disaster zone too dangerous for a human to enter, and get out before approaching a building. We name these stages *ingress*, *driving*, and *egress*, respectively. The morphology of the robot is not mandated, but we plan to use a humanoid (shown in Fig. 1; details in Section II). Furthermore, the vehicle to be driven is unknown but expected to be broadly similar to the ones in Figs. 1 and 5.

There are potentially an enormous number of perception, motion planning, and control problems to address in order for a robot to successfully complete this very practical challenge on a wide range of vehicles and roads. In this paper, we focus on only a handful of key perception tasks necessary during the *ingress* stage, with reference to the associated motion planning tasks that such scene understanding enables. The *driving* task is of course nontrivial, but our approach is algorithmically similar to previous work stemming from the various DARPA Grand Challenges from 2004-2007 [3], [4] and out of the scope of this work. Also, although *egress* is quite similar to *ingress*, the issues are more of motion planning and control rather than perception since vehicle parameters have already been estimated, so we will not examine it here.

We break the *ingress* perceptual task into several phases. Assuming the robot's initial pose is close to the vehicle and pointed toward it (so that search is not necessary), the first goal is to differentiate the driver's side and passenger side, and to find and parametrize a target area on the floor of the vehicle that the robot will be stepping up to. The height and lateral boundaries of this area are critical parameters for the motion planner. The second step of the ingress task is to parametrize the location and dimensions of the seat such that the robot can plan to safely lower itself down into a sitting position. If the robot has entered on the passenger side, it must then "scoot" laterally, possibly using its hands, to get into a driving position.

The last phase of ingress is something we term *interfacing*. During this phase control surfaces of interest such as the vehicle on/off switch, steering wheel, accelerator/brake pedals, and the gear shifter (for reversing) must all be located and parametrized. Furthermore, the robot must carry out a set of calibrations before the vehicle begins moving such as checking the reachability of these control surfaces, grasping/touching them, inferring or refining expected affordances, measuring force resistance, etc.

The main topic of this paper is a set of techniques for analyzing the 3-D structure of a vehicle in order to identify parts which are functionally important for ingress such as the *floor*, *seat*, *dashboard*, *steering wheel*, and *pedals*. With these areas labeled and parametrized, the robot has the information it needs to plan collision-free motions for stepping, sitting,

978-1-4673-6225-2/13/$31.00 ©2013 IEEE

scooting, grasping, and so on. However, for this paper the bulk of our attention is limited to two of these functional categories: the *floor* and the *steering wheel*.

As an important simplifying constraint, we currently assume that the utility vehicle has no roof or doors. The first of these would not significantly change the perceptual approaches presented here, but it would complicate motion planning by introducing the issue of head and shoulder clearance. Doors make floor estimation harder by partially obstructing the view of the vehicle interior, as well as requiring additional sensing and planning for grasping and opening.

In order to obtain 3-D information, we compare the efficacy of two different depth-sensing devices for these tasks: a Kinect-like stereo camera and a tilting laser range-finder. As discussed in Section II, these sensors have quite different strengths and price points. They both provide appearance information (either intensity or color) in addition to depth, but our emphasis here is completely on structural analysis.

A. Related Work

Closely related work includes efforts to get bipedal robots to step up or climb stairs. The Nao robot climbs a spiral staircase in [5] after stopping to acquire a point cloud with a short-range tilting Hokuyo and segmenting individual tread rectangles. The authors mention [6], [7] as key prior work on using RANSAC-like techniques to perceive a single step. The Honda Asimo robot sees isolated planar steps with a tilt ladar, stepping around or onto them in [8], [9]. There is also analysis of point clouds for stair perception in [10].

Several groups have been interested in 3-D object recognition more generally. [11] uses visual appearance, local shape and geometry, and geometric context features to label colored point clouds of indoor office and home scenes with a large number of classes such as *wall*, *floor*, *keyboard*, *tabletop*, *chairback*, *monitor*, *book*, and so on. [12] looks for objects such as cups, bowls, cereal boxes, etc. in point clouds with color information using an RGB-D variant of HOG detectors after first training on 3-D models. Similarly, much work has been done with the PR2 robot in terms of looking at tabletops and segmenting and identifying objects with its ladar and/or stereo cameras [13], including plane fitting and region growing for segmentation in a kitchen environment [14]. Also relevant is the work in [15] on door handle detection using a tilting Hokuyo after first finding doors using depth and reflectance information.

II. Equipment & Point Cloud Capture

As shown in Fig. 1, for this preliminary work we study a single full-scale golf cart with no roof or windshield. To obtain the pose of the robot or sensors during testing and data capture sessions, we have an optical motion capture system from OptiTrack which uses passive reflective markers.

Our robot is a humanoid called Hubo 2+ [16] which is 130 cm tall and has a mass of 42 Kg. It has 38 total degrees of freedom (DoF): 6 DoF in each limb, 3 DoF in the neck, 1 DoF at the waist, and 5 DoF per hand. Normal walking speed is 0.5 m/s, with a maximum of 1.0 m/s.

Fig. 2. Prototype sensor head with Asus Xtion Pro Live and tilting Hokuyo UTM-30LX-EW. Stereo color cameras, a PMD CamBoard nano, and a Microstrain IMU are also included but not used here.

The head shown in Fig. 1 does not contain any useful sensors for our task, so we have designed a prototype head, shown in Fig. 2. The head had motion capture markers affixed for all data collection, and it integrates the following two key sensors (along with several others not used in this work):

- **Asus Xtion Pro Live RGB-D camera** (a compact, low-power equivalent of a Microsoft Kinect) which captures RGB and depth images at 640×480 resolution with a field of view (FOV) of about $60° \times 40°$. The Asus has a maximum depth range of about 4 m and a minimum range of about 0.75 m, but it does not work in full sunlight.
- **Tiltable Hokuyo UTM-30LX-EW laser range-finder** which scans at 40 Hz over a $270°$ FOV at an angular resolution of $0.25°$. The minimum detectable depth is 0.1 m and the maximum is 30 m, and intensity-like reflectance information is provided for each point. The Hokuyo is mounted on a tilting servo which affords an unoccluded view from a minimum of $-90°$ (pointing straight down) to $+60°$.

Obtaining 3-D point clouds from the Asus is straightforward, as it furnishes depth images directly and is factory-calibrated with functions to obtain full X, Y, Z values for each pixel (through OpenNI). The Hokuyo ladar is tilting continuously in a sinusoidal pattern over a range of $[-45°, 45°]$ (relative to the sensor head pose) at a maximum speed of $10°$ / s. Each point cloud is assembled from the transformed laser scans over approximately one full sweep. To limit extraneous data for this task we retain only the front $180°$ of the FOV and remove all points beyond 4 m.

As an additional source of information, we use the `KinFu` module in the Point Clouds Library (PCL) [17], an open-source version of the original Kinect Fusion algorithm [1]. KinFu stitches together multiple views from the Asus and creates a smoother surface model of the vehicle through depth super-resolution than one frame alone contains. KinFu uses iterative closest point (ICP) on the 3-D point clouds captured

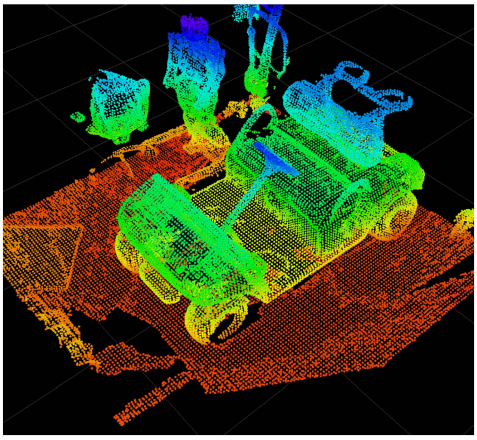

Fig. 3. Point cloud of golf cart scene captured from Asus depth camera with KinFu.

at successive sensor poses to estimate the camera motion and put all of the sequence's points into a common frame. ICP can fail when the scene does not have sufficient 3-D structure, as with large planar surfaces [18], but for the vehicle scanning we do it works quite well.

The point cloud obtained from KinFu for a handheld minute-long scan around the vehicle, voxelized at a resolution of 0.025 m and colored by height in ROS `rviz` [19], is shown in Fig. 3. In future work this point cloud will be gathered from the robot as it walks toward and around the vehicle.

All computations except the KinFu capture were done on an Intel i7-3720QM 2.6 GHz laptop with 16 Gb of RAM.

III. View Alignment

Throughout this paper we use the ROS [19] convention for coordinates of $+X$ pointing forward, $+Y$ to the left, and $+Z$ up.

A. Obtaining Vehicle Coordinates

As can be seen in Fig. 3, the KinFu-captured scene cloud includes points from the ground and other objects. Before further processing, we want to detect which points belong to the vehicle in order to filter out distracting background features and obtain the vehicle dimensions. As a first step, the ground plane is found and parametrized by a robust plane fit using RANSAC [17], the entire point cloud is *rectified* to put the ground plane at $Z = 0$, and ground plane points are removed by thresholding $z \leq 0.05$ m. A heightmap $\mathbf{H}_{scene}$ is then generated over a bounding box around the remaining KinFu scene points at a resolution of 0.01 m. $\mathbf{H}_{scene}$ is shown in Fig. 4(a) with red representing cells with no data and intensity proportional to z, up to a maximum of 1 m.

We formulate the vehicle detection problem as finding the position, orientation, and dimensions of a vehicle-sized rectangle $\mathbf{R}_{veh} = (x_{veh}, y_{veh}, \theta_{veh}, l_{veh}, w_{veh})$ in $\mathbf{H}_{scene}$. With appropriate bounds on l_{veh} and w_{veh} and a rough expectation of the average vehicle height, this is a well-posed problem if there is exactly one vehicle in the scene. For this work we learn bounds from published specifications of a representative

(a) (b)

Fig. 4. (a) 0.01 m resolution heightmap $\mathbf{H}_{scene}$ of KinFu-derived point cloud from Fig. 3 after rectification and ground removal. Red pixels are "no data" cells and intensity saturates at 1 m. (b) Estimated vehicle heightmap $\mathbf{H}_{veh}$

Fig. 5. Sample utility vehicles (Cushman Hauler and Kawasaki Mule) from set used to learn vehicle dimensions. Note the different heights and widths of the floor step areas and locations and angles of the steering wheels.

sample of 7 similar utility vehicles.[1] Minimum and maximum values of the vehicle *length*, *width*, and *aspect ratio* were computed over this set, and these were scaled down and up by 90% and 110%, respectively, to get absolute bounds.

The likelihood $P_{veh}(\mathbf{R})$ of a particular hypothetical rectangle $\mathbf{R} = (x, y, \theta, l, w)$ is measured via height contrast: intuitively, we are looking for a rectangle-shaped region filled with obstacle points surrounded by some amount of free space in $\mathbf{H}_{scene}$. We quantify this by counting the number of occupied cells O_{in} and free cells F_{in} inside $\mathbf{R}$, the number of occupied cells O_{frame} in a rectangular *frame* around $\mathbf{R}$, and compute the contrast $P_{veh}(\mathbf{R}) = O_{in} - w(F_{in} + O_{frame})$. For the results here $w = 0.5$ and the frame width is 0.2 m. Note that there is no distinction made between the front end and the rear end of the vehicle. This ambiguity is resolved at the part detection stage in Section IV.

To find the maximum likelihood, we run a particle filter [20] with 200 particles, starting with a uniform prior distribution on the state variables within the learned dimensional bounds and the positional bounds of $\mathbf{H}_{scene}$ (θ is completely unknown). The state after 250 iterations is taken as the best vehicle rectangle $\hat{\mathbf{R}}_{veh}$. If a randomly-generated hypothesis violates the dimensional bounds given above, rather than assigning it a zero likelihood we simply resample it from the prior distribution.

The heightmap resulting from the vehicle estimate, which takes a few seconds to obtain, is shown in Fig. 4(b). The camera path did not get full coverage on the passenger side, so

[1] Kawasaki Mule 4000, Cushman Hauler 800 electric, Polaris Ranger EV, Deere R-Gator, Deere HPX 4x4, Honda Big Red, and Bobcat 4200 4x2

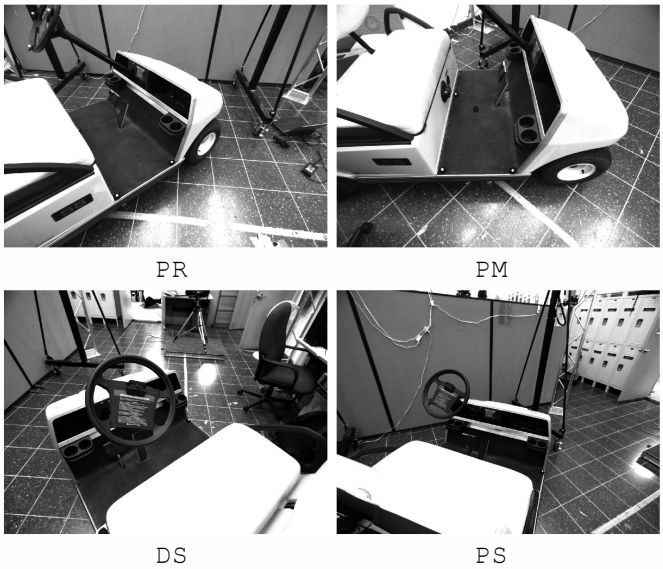

Fig. 6. Right stereo camera images from static capture locations described in text

the point cloud is ragged there, resulting in a slightly wrong width estimate. θ is also slightly off, most likely due to the low resolution of the heightmap. There are several ways to improve both of these issues with additional processing, but these results are sufficient for the other perceptual tasks that need them.

B. Registration of Individual Views

Part detection can operate on the full vehicle heightmap $\mathbf{H}_{veh}$ in Fig. 4(b), and we show results for doing so in Section IV. However, we are also interested in detecting parts in the sensor point clouds obtained at individual poses as the robot moves. With a sensor's limited FOV and intra-vehicle occlusions, parts such as the floor or steering wheel might be completely out of view or only partially visible.

To illustrate this, data on the test vehicle was captured from four static positions with the sensor head on a tripod, all at about the head height of the Hubo. Views from the right stereo camera are shown for these positions in Fig. 6. In `PassengerMid (PM)` the vehicle floor is in view but the steering wheel is out of frame. In `PassengerRear (PR)` the floor is mostly visible and the steering wheel is partially visible. In `DriverSteering (DS)` the steering wheel is prominently in view but the floor is partially blocked by the seat. Finally, in `PassengerSteering (PS)` the steering wheel is in view but the floor is almost completely occluded.

Each sensor point cloud is rectified and ground plane points removed as described above for the KinFu data. The procedure is somewhat different because there may be little or no ground visible in close-up views of the vehicle. Thus, as a first step the point cloud is *roughly* rectified using the height of the sensor head and its tilt angle with respect to the ground as reported by the lab's motion capture system described in Section II.[2] Rough ground points ($|z| \leq 0.1$ m) are then selected; if there are none, this is the final ground plane. Otherwise, RANSAC robust plane-fitting with an inlier threshold distance of 0.01 m followed by least squares refinement is applied to the rough ground points to obtain the final ground plane. This plane is used to rectify the original sensor point cloud and remove ground points with a threshold of 0.05 m. Assuming that the sensor is near the vehicle and pointed at it, we also remove all points more than 2 m away. The rectified point clouds at pose `DS` from the Asus and Hokuyo before ground point removal and distance filtering are shown in Fig. 7.

[2]In the field, the tilt angle would be supplied by the IMU and the height of the sensor head would be derived from the robot kinematics

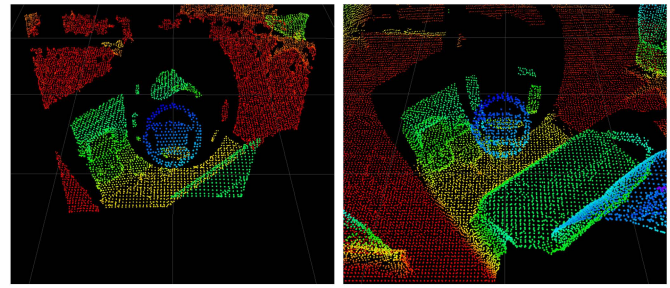

Fig. 7. Cropped views of point clouds obtained from `DS` pose in Fig. 6 after rectification

Even after rectification and filtering, interpreting these sensor point clouds can be difficult because they only show a portion of the vehicle, and *which* portion is unknown. We remove this uncertainty by attempting to register each sensor point cloud to the full KinFu vehicle point cloud. Because both clouds are rectified, it is only necessary to find a 2-D translation and rotation $\mathbf{T} = (\Delta x, \Delta y, \Delta\theta)$. Rather than work with the point clouds, it is efficient to convert the sensor cloud for sensor s to a heightmap $\mathbf{H}_s$. Sample sensor heightmaps for the Asus and Hokuyo at pose `DS` derived from the point clouds in Fig. 7 are shown in Fig. 8. Note that these are different sizes because the sensors' different FOVs result in different bounding boxes. Their scales are the same, as are their orientations.

Given the non-vehicle points which may be present in $\mathbf{H}_s$, finding the $\hat{\mathbf{T}}$ which makes it best agree with $\mathbf{H}_{veh}$ is essentially a robust image template-matching problem. A standard approach would be to compute features such as SIFT, SURF, etc. in each heightmap "image", match them, and estimate $\mathbf{T}$ from the matches in a RANSAC-like framework, but this is complicated here by the small size of the images and the "no data" cells/pixels. For this preliminary work we found that a simple, successful approach is to formulate a pixelwise objective function $f(\mathbf{T}, \mathbf{H}_s, \mathbf{H}_{veh})$ which measures the degree of match between $\mathbf{H}_{veh}$ and $\mathbf{H}_s^{\mathbf{T}}$, the sensor heightmap after transformation by $\mathbf{T}$, and set $\hat{\mathbf{T}} = \mathrm{argmax}_{\mathbf{T}} f(\cdot)$. To evaluate $f()$, we iterate over all pairs of corresponding heightmap cells $(h_{veh}, h_s^{\mathbf{T}})$ in the overlapping portion of $\mathbf{H}_{veh}$ and $\mathbf{H}_s^{\mathbf{T}}$ and count the number of *matches*. A pair is considered a match if (a) there is height data for both heightmap cells (i.e., neither one is red) and (b) their heights are relatively close–we use $|h_{veh} - h_s^{\mathbf{T}}| \leq 0.1$ m.

To optimize $f()$, we do an exhaustive search at a quarter of the original heightmap resolution at 1 pixel translational and 1° angular increments to find an approximate solution $\hat{\mathbf{T}}_{1/4}$, then

Fig. 8. Sensor heightmaps of point clouds obtained from DS pose in Fig. 6 after rectification and ground removal

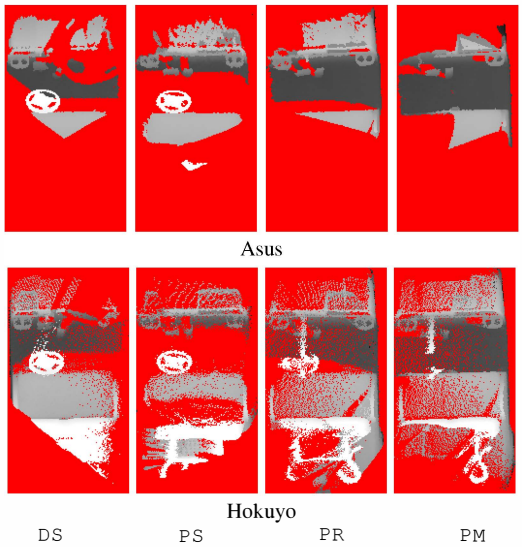

Fig. 9. Registered and trimmed heightmaps of individual sensors

search again at half-resolution within a tighter angular range of $\Delta\hat{\theta}_{1/4} \pm 5°$ at $0.5°$ increments for a better solution $\hat{\mathbf{T}}_{1/2}$. This process takes about 1 minute to compute. The full sensor cloud is then transformed with $\hat{\mathbf{T}}_{1/2}$ and ICP is performed between it and the KinFu vehicle cloud to get a final $\hat{\mathbf{T}}$. The registered sensor heightmaps found for each of the four example views (after trimming points outside the vehicle bounds), which we will term $\mathbf{H}_s^{\mathtt{DS}}$, $\mathbf{H}_s^{\mathtt{PS}}$, $\mathbf{H}_s^{\mathtt{PR}}$, and $\mathbf{H}_s^{\mathtt{PM}}$, are shown in Fig. 9.

IV. Vehicle Part Detection

With a sensor view registered to the vehicle and non-vehicle points trimmed away, the search for any parts of interest to the motion planner is considerably constrained. We detail these constraints and the individual part detectors below.

A. Floor

Intuitively, the floor is expected to be a planar rectangular region parallel to the ground plane. Unfortunately, we cannot simply use RANSAC to find the strongest horizontal plane in a particular registered sensor heightmap $\mathbf{H}_s^{\mathbf{T}}$ because of the potential for confusion with the seat or hood planes, as can be seen from Fig. 9. However, we have prior knowledge that the floor plane must be at a steppable height. Assuming that this is a roughly constant offset above the vehicle's ground clearance, we once again use specifications from the set of exemplar vehicles introduced in Section III-A to obtain minimum and maximum bounds on the floor height (in this case $z_{floor} \in [0.15, 0.40]$ m). Excluding points in the registered sensor point cloud outside of this range yields a nominal *floor slice* upon which we then run a RANSAC horizontal plane fit (i.e., the normal must be within $5°$ of vertical). This finds the floor plane in seven of the eight $\mathbf{H}_s^*$ in Fig. 9 as well as $\mathbf{H}_{veh}$ in Fig. 4(b) (not enough inliers were found in $\mathbf{H}_{hok}^{\mathtt{PS}}$).

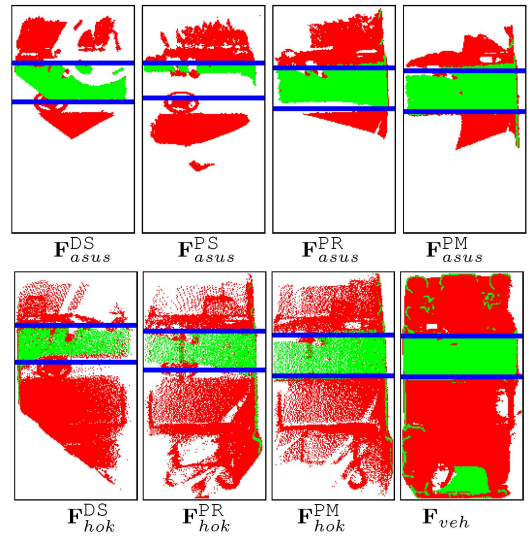

Fig. 10. Inliers after horizontal plane fitting on floor height slice in green, outliers (all other vehicle points) in red, and fitted rectangles outlined in blue. "No data" points are white here, and inliers overwrite outliers for display (which is why the steering wheel is oddly cropped in $\mathbf{F}_{asus}^{\mathtt{DS}}$)

Fig. 10 shows the floor plane inliers for each sensor and view $\mathbf{F}_s^*$ in green and outlier points from the vehicle in red. Isolating the cluster of inliers belonging to the floor region itself can be formulated as a rectangle finding problem similar to the vehicle detection task in Section III-A. However, whereas that problem has a 5-D search space, this one is more constrained. We assume that the floor rectangle's axes are aligned with those of the vehicle, its width is approximately the same as the vehicle, and its center is on the vehicle centerline. This leaves only 2 free variables to determine $\mathbf{R}_{floor}$: (x_{floor}, l_{floor}), the floor's forward/backward position and dimension (i.e., the distance between the seat and the dashboard). We put reasonable bounds on these variables and again run a particle filter with the likelihood function $P_{floor}(\mathbf{R}) = (N_{in} - N_{out})/A$, where N_{in} is the number of floor plane inliers in $\mathbf{R}$, N_{out} is the number of outliers in the rectangle, and A is its area.

The search is very fast to converge and only 10 iterations are needed. The blue lines in Fig. 10 indicate the estimated floor rectangles. These are accurate when the whole floor region is visible, and conservative when it is not. Impingements like the pedals, drink holders, and steering wheel are detectable as outlier points present inside the floor rectangle, and this information can be passed on to the motion planner. Some error is due to rotational inaccuracy in the registered sensor heightmaps.

B. Steering Wheel

For purposes of pattern recognition, the steering wheel is essentially a circle on an inclined plane. From automotive standards there are fairly tight bounds on the possible radius r, and we expect that it will be tilted in the range of $\phi \in [20°, 70°]$,

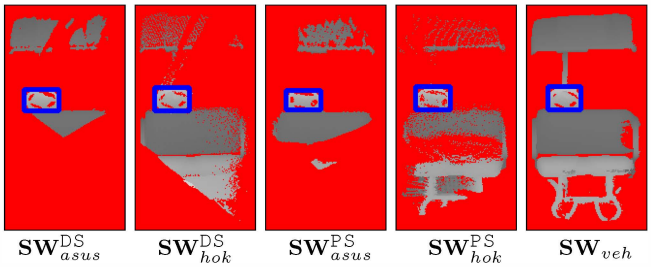

Fig. 11. Fitted steering wheel rectangles outlined in blue on steering wheel height slice ("no data" points are red here)

where $0°$ would be parallel to the ground plane. As before, we can limit the search for it to a reasonable range of heights to filter out some vehicle points, and its x, y position within the vehicle is somewhat constrained (even not knowing front from back or whether it is a right- or left-hand drive vehicle).

Our approach here is to interpret a particular steering wheel pose and size hypothesis in terms of an axis-aligned (in vehicle coordinates) bounding rectangle $\mathbf{R}_{sw}$ in a registered sensor heightmap $\mathbf{H}^*_s$. This rectangle's center is defined by x_{sw}, y_{sw}, its width is just $2r_{sw}$, and its length is $2r_{sw}\cos\phi$. Within a hypothetical bounding rectangle we expect to see an ellipse of points whose heights contrast with those of nearby points just outside the ellipse (i.e., floor and/or seat points). We quantify this by sampling the ellipse at N discrete points and counting how many pairs of inside/outside points N_{in} have a height difference of ≥ 0.05 m or the outside point has no data vs. how many N_{out} are about the same height. The likelihood of the associated rectangle is then $P_{sw}(\mathbf{R}) = (N_{in} - N_{out})/N$.

A particle filter is again used to search for 200 iterations over different rectangles to optimize P_{sw}. If the likelihood of the best rectangle found is less than 0.5, we say that no steering wheel has been found. With this criterion, all of the detections are shown in Fig. 11. The estimated diameters range from 0.353 to 0.357 m; the hand-measured diameter of the golf cart steering wheel is 0.346, a 1 cm difference.

V. Conclusion

We have presented several techniques for detecting and estimating parameters of utility vehicle parts as prerequisites for for humanoid robot ingress. More polishing could be done to improve the accuracy of the estimates, but the basic methods are robust for both sensors tested. In future work we will include other part types such as the seat and the pedals, and refine our knowledge of the steering wheel's spoke locations and tube diameter so that it can actually be gripped. Thus far the detectors presented are "first-order", in that they do not depend on or otherwise exploit one another's output. We are currently examining how the knowledge of where the floor is may help steering wheel detection and vice versa, as well as all of the other categories under consideration.

Demonstrating generality is very important, and we plan to use KinFu to collect models of a number of other utility vehicles to test and refine these techniques. We are also working on integrating the sensor head with Hubo and getting these methods to run in real-time. Incremental registration of the sensor head to the KinFu point cloud as it is built should be much more efficient than the approach given in Section III-B.

Acknowledgments

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) award #N65236-12-1-1005 for the DARPA Robotics Challenge.

References

[1] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," in *ACM Symposium on User Interface Software & Technology*, 2011.

[2] DARPA, "DARPA Robotics Challenge website," available at `http://darparoboticschallenge.org`. Accessed December, 2012.

[3] S. Thrun, M. Montemerlo *et al.*, "Stanley, the robot that won the DARPA grand challenge," *Journal of Field Robotics*, vol. 23, no. 9, 2006.

[4] C. Urmson *et al.*, "Autonomous Driving in Urban Environments: Boss and the Urban Challenge," *Journal of Field Robotics*, vol. 25, no. 1, 2008.

[5] S. Oßwald, J. Gutmann, A. Hornung, and M. Bennewitz, "From 3D Point Clouds to Climbing Stairs: A Comparison of Plane Segmentation Approaches for Humanoids," in *Proc. of the IEEE-RAS Inter. Conf. on Humanoid Robots*, 2011.

[6] K. Okada, S. Kagami, M. Inaba, and H. Inoue, "Plane segment finder: Algorithm, implementation and applications," in *Proc. IEEE Inter. Conf. on Robotics & Automation*, 2001.

[7] J. Gutmann, M. Fukuchi, and M. Fujita, "Stair climbing for humanoid robots using stereo vision," in *Proc. IEEE/RSJ Inter. Conf. on Intelligent Robots & Systems*, 2004.

[8] J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, and S. Kagami, "Biped navigation in rough environments using on-board sensing," in *Proc. IEEE/RSJ Inter. Conf. on Intelligent Robots & Systems*, 2009.

[9] K. Nishiwaki, J. Chestnutt, and S. Kagami, "Autonomous navigation of a humanoid robot over unknown rough terrain using a laser range sensor," *Inter. Journal of Robotics Research*, vol. 31, no. 11, pp. 1251–1262, 2012.

[10] V. Sanchez and A. Zakhor, "Planar 3D Modeling of Building Interiors from Point Cloud Data," in *ICIP*, 2012.

[11] H. Koppula, A. Anand, T. Joachims, and A. Saxena, "Labeling 3D scenes for Personal Assistant Robots," in *Proc. RSS workshop on RGB-D Cameras*, 2011.

[12] K. Lai, L. Bo, X. Ren, and D. Fox, "Detection-based Object Labeling in 3D Scenes," in *Proc. IEEE Inter. Conf. on Robotics & Automation*, 2012.

[13] Z. Marton, D. Pangergic, R. Rusu, A. Holzbach, and M. Beetz, "Hierarchical Object Geometric Categorization and Appearance Classication for Mobile Manipulation," in *Proc. IEEE-RAS Inter. Conf. on Humanoid Robots*, 2010.

[14] R. Rusu, Z. Marton, N. Blodow, A. Holzbach, and M. Beetz, "Model-based and Learned Semantic Object Labeling in 3D Point Cloud Maps of Kitchen Environments," in *Proc. IEEE/RSJ Inter. Conf. on Intelligent Robots & Systems*, 2009.

[15] R. Rusu, W. Meeussen, S. Chitta, and M. Beetz, "Laser-based Perception for Door and Handle Identification," in *Proc. Inter. Conf. on Advanced Robotics*, 2009.

[16] R. Ellenberg, R. Sherbert, P. Oh, A. Alspach, R. Gross, and J. Oh, "A common interface for humanoid simulation and hardware," in *Proc. IEEE Inter. Conf. on Humanoid Robots*, 2010.

[17] R. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Inter. Conf. on Robotics & Automation*, 2011.

[18] T. Whelan, J. McDonald, H. Johannsson, M. Kaess, and J. Leonard, "Robust Tracking for Dense RGB-D Mapping with Kintinuous," in *To appear in Proc. IEEE Inter. Conf. on Robotics & Automation*, 2013.

[19] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: An Open-Source Robot Operating System," in *Proc. ICRA workshop on Open-Source Software*, 2009.

[20] A. Blake and M. Isard, *Active Contours*. Springer-Verlag, 1998.