Full-Speed Jackal

(A project of the 2017 Robotics Course of the School of Information Science and Technology (SIST) of ShanghaiTech University)

Instructor: Prof. Sören Schwertfeger · https://robotics.shanghaitech.edu.cn/teaching/robotics2017

Kuang Haofei[1], Wang Lei[2], Li Ziyue[1] and Zeng Xiangchen[1]

[1]School of Information Science and Technology(SIST), ShanghaiTech University
[2]Shanghai Advanced Research Institute, Chinese Academy of Sciences

***Abstract*—Running Jackal robot in the full speed state has to satisfy two types of constrains: *obstacle* and *robot model*. It couldn't be satisfied with current ROS navigation package, like *move_base*. Address this problem, we implemented the *Model Predict Controller(MPC)* into Jackal robot to perform local navigation task. It could make Jackal robot to run with it's full speed in a static environment without any obstacle collisions. But the speed of computing optimal result in mpc is slower than our expectation, so we change to PID method which doesn't need consider the velocity of obstacles and improved computation speed significantly. The implementation of PID is mainly about to compute a primary direction based on the laser data and switch in four states to make efficient PID controllers. We achieved our goals of letting Jackal run in a multi obstacles environment with high speed, it can detect obstacles and avoid obstacles automatically. And we also will implement *leg detection* algorithm to detect and track human, then estimate their speed for predicting their position.**

I. Introduction

This project focus on running Jackal robot with full speed in a dense obstacle environment. We implement the PID controller as the strategy of this project. And we also tried other methods such as MPC which was abandoned because of online speed restrictions.

Then, we need find a high efficient procedure of segmentation, tracking and the speed estimation of the moving objects. Full-speed Jackal is a challenging but interesting project because it combine the robot with the real-time situation planning and analyzing. By doing this project, we can get in touch with some fancy algorithms and meaningful ideas, it help a lot for our further study of the mobile robots.

II. State of the Art

We read some paper which is related to this project, and understanding some state of the art technology of PID controller, MPC and human tracking. In this section, we will discuss these paper in detail.

A. PID controller

PID controller is a proportional–integral–derivative controller. It is a control loop feedback mechanism widely used in industrial control systems and a variety of other applications requiring continuously modulated control. A PID controller continuously calculates an error value $e(t)$ as the difference between a desired setpoint and a measured process variable and applies a correction based on proportional, integral, and derivative terms (denoted P, I, and D respectively).

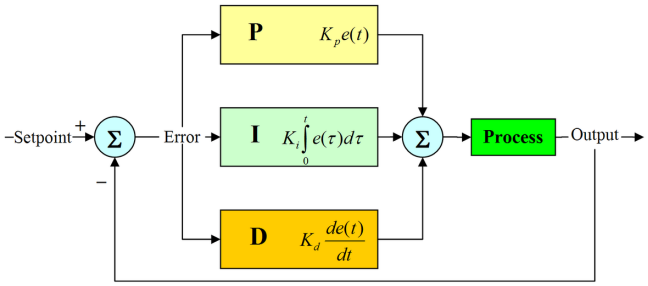

Fig. 1: Schematic of the MPC-based obstacle avoidance algorithm

In this case, we could calculate the error between our desired direction and robot direction according to robot dynamic model. Then, using PID controller to compute control signal and make robot move.

B. Model Predict Controller

Model predictive control (MPC) is an advanced method of process control that has been in use in the process industries. Over a past decade MPC is used to address a wide range of real world problems. This success is motivated due to deal easily with constraints on both state and control variables during the evolution of the system under control. It also be used in robot obstacle avoidance at these years.

1) Occupancy Grid based MPC for Mobile Robot: There is a paper [1] implement distributed MPC for non-holonomic mobile robots[1]. The key idea of the paper is to project the predicted state trajectory onto a grid resulting in an occupancy grid prediction. They implement MPC scheme to stabilize a group of autonomous non-cooperative differential driver robots. In MPC, the first thing is measure the current state of the system, which serves as a basis for solving

a finite horizon optimal control problem(OCP). The OCP cost function allows us to encode a control objective and to evaluate the closed-loop performance. Then, the first element of the computed sequence is applied before the process is repeated. This method is applied in multi-robot system for making a robot to avoid another robot. each robots using same policy. First, partition the operating region into a grid. Then, the predicted trajectories are projected onto the grid resulting in an occupancy grid prediction, which serves as a quantization of the communication data. Based on an exchange of these projections, each robot formulates suitable coupling constraints.

This paper using occupancy grid based MPC to reduce the communication load, but the application is very similar with our project. So, we can use the method to build a system for making robot avoid human in dynamic environment. When we get the position and velocity information of peoples from out people tracking module, we could predict which grid will be occupied and adding the constraint to MPC. We can got a accuracy trajectory which avoid every occupied grid. MPC also compute the control signal to make robot move.

2) Control and Obstacle Avoidance of Mobile Robot: This paper [2] mainly concern at how to use MPC controller combined with the feedback in controling the obstacle avoidance. The proposed algorithm is based on the concept of robust MPC control [3] [4] that guarantees the single robot to avoid collision in the presence of only fixed obstacles. So as we considering the moving obstacle at a later stage, we will refer to this work for fixed obstacle, and applied to moving obstacle based on modification.

Jackal kinematics is basicially same as the mobile robots used in this paper.

The goal of this paper is to build a control technique which drive the robot to the desired position without a phase of planning the entire route. And the corresponding MPC-based obstacle avoidance algorithm is as following.

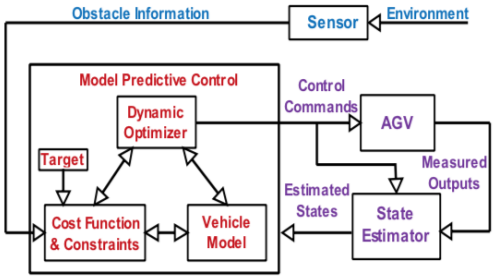

Fig. 2: Schematic of the MPC-based obstacle avoidance algorithm

The idea is to use an appropriate local control loop to manage the movement of the robot at a precise reference and use an external control loop based on predictive techniques to generate the best orientation towards the completion of the task, remaining at a distance from obstacles. In this regard the predictive approach for the optimization problem [5] is used, limited horizon considered, obstacle avoidance problems typical of path planning phase of any autonomous navigation problem in the generic sense is considered.

It should be noted that we have to use feedback control in this problem because the execution of the same task is affected by uncertainty.

And after get the MPC problem formulation, consider the stability condition for Jackal is also very important.

3) Open-source Nonlinear optimization library: NLopt: For MPC of mobile robot module, it doesn't has any open-source ROS package now. So, we need to writing a ROS package to solve the problem by ourselves. MPC need to solve a optimization problem, so we need to use some optimization algorithm in the project. But solve the problem need to solve too much mathematic problems, it is a huge work. So, we need to use some open-source solver to achieve the problem. NLopt is the good library of it.(https://nlopt.readthedocs.io/en/latest/)

NLopt is a free/open-source library for nonlinear optimization, it provide a common interface for a number of different free optimization routines available online as well as original implementations of various other algorithms. It could be used in our MPC easily. So, our task is converted to model our problem and provide constraint to it.

C. Human Tracking and Speed Estimation

1) Related Work: The primary objective of this research is to promote the development of robust, repeatable and transferable software for robots that can automatically detect, track and follow people in their environment. The authors' work strongly motivated by the need for such functionality onboard an intelligent power wheelchair robot designed to assist people with mobility impairments. In this paper the authors proposed a new algorithm for robust detection, tracking and following from laser data. They showed that the approach was effective in various environments, both indoor and outdoor, and on different robot platforms (the intelligent power wheelchair and a Clearpath Husky). The method has been implemented in the Robot Operating System (ROS) framework and will be publicly released as a ROS package.

2) One Open-source ROS Package:leg_tracker: As we presented another open-source ROS package *leg_detector* in project proposal, it takes *sensor_msgs/LaserScans* as input and uses a machine-learning-trained classifier to detect groups of laser readings as possible legs. But, this *leg_detector* only provide individual legs model constantly rather than adapt to Jackal. To solve this issue, we need retrain a leg model that suitable for Jackal.We found a paper [6] in ICRA 2015 relate to train a leg model which uses a machine-learning classifier, and the author publicly released his retraining method as a ROS package. This ROS package is also based on the *leg_detector*, and did improve for robust detection and tracking. As for the machine-learning algorithm proposed in this package: firstly, scan points returned are first clustered according to a fixed threshold, such that any points within the threshold are grouped together as a cluster and then further classified as human or non-human; secondly, it used a OpenCV's random forest [7] classifier to

do the training based on rich positive/negative examples, one benefit of using an ensemble classification method, which included in the random forest classifier, is that a measure of confidence in the classification can be extracted.

III. System Description

We implemented algorithms which is presented on last section and also PID. Firstly, we need to build a simulation environment for testing our algorithm, it should similar to our real robot as much as possible. A navigation task need achieve two tasks: *Go to Goal* and *Avoid Obstacle*. For achieve these two tasks, we design a PID controller by using odometer data and Velodyne data. Then we implement PID into Jackal robot, it should make robot move with full speed and avoid every obstacle in a multi obstacles environment. And we also will talk about implemented but abandoned MPC algorithm. Beyond that, the human tracking task part. In this section, we will discuss these algorithms in detail.

A. Jackal Kinematics

The kinematics equations that govern the motion of the robot is as given below:

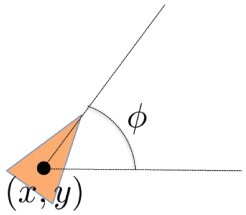

Fig. 3: Jackal model

$$\begin{cases} \dot{x} = v\cos\theta, \\ \dot{y} = v\sin\theta, \\ \dot{\theta} = \omega, \end{cases} \tag{1}$$

We build the dynamic model of Jackal as a unicycle model, that means control signal is linear velocity v and angular velocity w.

B. Jackal Simulation

The simulation environment is very important for our project, especially testing PID controller and MPC of mobile robot. Because robot will move according to our algorithm, so we have to make sure our algorithms is perfect before we test it in real robot. So, we build a simulation environment of Jackal with a Velodyne VLP-16 for testing performance of algorithm.

Our robot is a Clearpath Jackal robot, and a Velodyne VLP-16 laser scanner is mounted on it. We have to make our model similar with our real robot, so we combine with the Jackal simulation ROS package [8] and Velodyne simulation ROS package [9] in the Gazebo environment. The layout of simulator is same as our real robot and sensor, so we could testing our algorithm in simulation and transfer our algorithm into real robot with some slight modify directly. The simulation is show as Fig 4.

Fig. 4: Gazebo simulation of Jackal robot with Velodyne VLP-16.

C. Go to Goal

With regard to a navigation task, the original target is make robot could move to a destination. So, go to goal is a basic task for the project. In this case, we just consider local navigation, that means we don't have map and could not use any advanced planning algorithm. So, we use odometer frame as our world frame and we could get the pose $P(x, y, \theta)$ of robot from the odometer data. For destination, we could use a relative position of robot, and transform it to world frame.

We use a PID controller to compute control signal of robot. Our strategy is set linear velocity(v) as a constant and compute the error of direction, then computing the control signal by using PID controller. First of all, we need to compute the error. We make the destination as an vector in the world frame and compute angular of the vector. Then, setting error as subtract the angular of destination and the head of robot. As given below:

$$e = \theta_d - \theta \tag{2}$$

It's worth noting that we have to make sure e is in $[-\pi, \pi]$. After doing it, we need to use PID controller to compute the angular velocity(w) of robot in this time. So, we have:

$$\omega = K_P e(t) + K_I \int_0^t e(\tau)d\tau + K_D \frac{de(t)}{dt} \tag{3}$$

In formula (2), $e(t)$ is the error in t time, K_P, K_I and K_D is the coefficients of PID controller. So, we need to use some way to compute the integral and differential of error. We could discrete time, and use an approximate solution. It is given below:

$$\Delta t = 1/f$$

$$\frac{de(t)}{dt} \approx \frac{e_t - e_{t-\Delta t}}{\Delta t} \tag{4}$$

$$\int_0^t e(\tau)d\tau \approx \sum_{k=0}^{N} e(k\Delta t) =: \Delta t E$$

$$\Delta t E_t = \Delta t \sum_{k=0}^{N+1} e(k\Delta t) = \Delta t e((N+1)\Delta t) + \Delta E_{t-\Delta t}$$

$$E_t = E_{t-\Delta t} + e\Delta t \tag{5}$$

Here, f is the frequency of the program. In this case, f is 20Hz, so $\Delta t = 0.05$s. So, PID controller will decrease the error between the head direction of robot with direction of destination in each move.

D. Avoid Obstacle

Avoid obstacle is another part of navigation task. We want to make robot go to a destination without crashing any obstacles, so the robot should have ability to avoid obstacle. In this case, we learning from the idea of artificial potential field method implement an algorithm for avoiding obstacle by just using laser scanner data as input.

First, we have to process raw laser data. Our laser scanner could get environment data with 360 degree, but we just use the data of front 180 degree, i.e. $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Because our robot model just need go forward and could not go back, we just consider to obstacle of front scene. It seems like Dubins path, but not identical to it, because our model could turn in place. The horizontal resolution of Velodyne is 0.4 degree, that means we could get 898 pointcloud each scan, we pick the front half of it is 499 pointcloud. In this case we don't need as much pointcloud, so we use a ROS package *pointcloud_to_laserscan* [10] convert pointcloud data to laser scan data with a 1 degree resolution.

The raw laser data just have the distance data, so we need to transform it to world frame as a vector. We could use transforms matrix to calculate the it:

$$\begin{bmatrix} x_{l_i} \\ y_{l_i} \\ 1 \end{bmatrix} = {}^{W}T_{R}\,{}^{R}T_{V} \begin{bmatrix} d_i \\ 0 \\ 1 \end{bmatrix} \tag{6}$$

In formula (5), d_i is the distance of i-th beam, ${}^{W}T_{R}$ is transform matrix from world frame to robot frame and ${}^{R}T_{V}$ is transform matrix from robot frame to Velodyne frame. And the transform matrix could get from ROS *tf* .So, we could get i-th laser beam vector $[x_{l_i}, y_{l_i}]^T$ in the world frame. Then, we subtract it with robot current position $[x_r, y_r]^T$:

$$\mathbf{v}_{d_i} = \begin{bmatrix} x_{d_i} \\ y_{d_i} \end{bmatrix} = \begin{bmatrix} x_{l_i} \\ y_{l_i} \end{bmatrix} - \begin{bmatrix} x_r \\ y_r \end{bmatrix} \tag{7}$$

$$\mathbf{v}_d = \sum_{i=0}^{n} \mathbf{v}_{d_i} \tag{8}$$

In formula (6), $\mathbf{v}_{d_i}$ is the i-th beam vector of relative to robot. Now, we sum up the vectors, and direction of the result $\mathbf{v}_d$ is the target angular. Like Fig 5. Now we could calculate θ_d of the vector, and take it to formula (1) and (2), we could get a angular velocity w. By the way, we also setting the linear velocity as a constant in this case. So, if there is no obstacle, the direction of result of sum up all laser vectors is robot head direction. In the situation, robot just go straight.

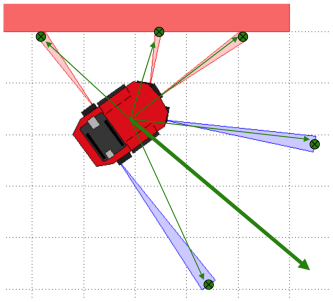

Fig. 5: Calculate the vector by sum up all laser vectors.

E. Finite State Machine

Our algorithm basically use state machine transition to switch in four states. Like following graphic expression:

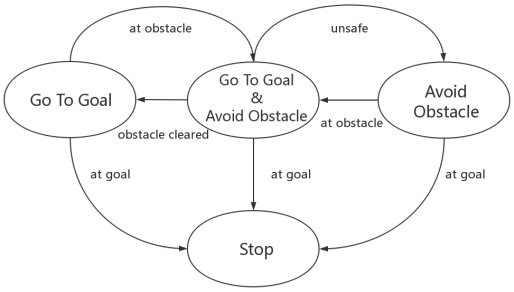

Fig. 6: State machine

1) state machine and state verification: So we will judge which states robot is in every frame, then consecutively switch to corresponding PID controller where each controllers have different parameters for respective efficiency. And we also set the dangerous distance value, if the minimum distance is less than it robot will only have angular velocity but zero linear velocity.

The judgement is based on the obstacle distance information given by laser. 'at goal', if the coordinate of destination and our current coordinate is reasonably close enough. 'obstacle cleared', if the minimum distance is larger than threshold value, we think the cloest obstacle is far enough to ignore now. 'at obstacle', if the minimum distance is less than some threshold value, so we need to consider avoid obstacle and go to our goal at the same time. 'unsafe', if the minimum distance is less than threshold value which even much less the threshold value of 'obstacle cleared', so robot now is too close to obstacles, need to avoid first.

F. Model Predict Controller

For build a MPC for Jackal robot, we divide the task to several tasks. First thing is learning the fundamental knowledge of MPC. Then, building a framework of MPC for our robot without obstacle, and make sure it could reach the destination. Then last thing is adding obstacle and dynamic human as constraints.

First, we solve a linear constraint and a nonlinear constraint optimization problem by using MATLAB and NLopt. Then, we solve a classical optimal control problem(OCP), because MPC is consist of several OCP.

After learning, we need model our robot for MPC. Because our robot is a unicycle model, it is a nonlinear system. So, we use 4th order Runge-Kutta method to get the numerical solution of the robot state. Then we build a one step MPC for our robot, adding the numerical solution to MPC as robot model constraint. The state is a 5 dimension vector $(x, y, \theta, v, \omega)$, v and ω is our control signal. And our cost function is((x^*, y^*) is except position):

$$cost = \sum_{k=0}^{N-1} (x_k - x^*) + (y_k - y^*) \tag{9}$$

When we running the program, we could get the right trajectory and a set of (v, ω). But unfortunately, the running speed is very slow. After we analyzing the problem, we found the program spend a lot of time on Runge-Kutta method. So, we need to use other method to solve the robot model or compute the analytic solution of unicycle model to achieve the problem.

G. Human Tracking and Speed Estimation

The second step problem is making Jackal can planning in dynamic environment, which considering walking pedestrians mainly. This problem of interest can be decomposed into four sub-problems: (1)segmentation, (2)detection, (3)tracking, (4)velocity estimation.

As our tracking system process, it solves above corresponding problems. Detected clusters are used as observations for the tracker. The observation at timestep k are denoted as $z_k = \{z_k^1, z_k^2, ..., z_k^{M_k}\}$ where M_k is the total number of detected clusters at time k. Firstly, to solve the issues of segmentation and detection, we used machine-learning method random forest to train classifier. Its features are: *Number of points, Standard deviation, Linearity, Boundary length, Mean angular difference, Width, Circularity, Boundary regularity, Mean curvature, Distance etc*. We recorded 6 positive rosbags and 6 negative rosbags by Jackal, with the help of the training algorithm above mentioned, we trained Jackal's own classification with 6410 positive and 1553 negative samples. Next step, we are using Kalman filter tracking and GNN data association for multi pedestrians tracking. If we could make our tracking part work efficiently, than we can mark each pedestrian detected and track them constantly. As we finally developing all above process, it will output a vector with position, velocity, orientation messages to Jackal.

To keep the number of person tracks manageable, person track initiation conditions and deletion criteria should be set in advance. In our method initiation conditions are applied as (1)a pair of tracks are detected, which move a given distance (0.5m) without drifting apart, (2)both tracks maintain a confidence level above the threshold. Finally, human tracks are deleted when their innovation covariance is greater than the threshold, or confidence over the track drops below the threshold.

For tracking multiple people, the Kalman filter for each track has a state estimate with the position and velocity of the cluster in 2D coordinates. Global nearest neighbor(GNN) data association is solvable in reducing uncertainty pertaining to how detections should be matched to tracks from the previous time to produce updated tracks for the current time, since the system is designed to track all detected scan clusters. We also integrate local occupancy grid mapping to improve data association by disallowing the initiation or continuation of people tracks in occupied space. As we firstly described a package *leg_detector* in our proposal, we also used the method that track all detected clusters in every frame, included non-human clusters.

IV. System Evaluation

A. Model Predict Controller

We did some simulation in obstacle cleared environments to evaluate our algorithm.

1) Experiment
   a) Test Jackal robot in simulation environment with static obstacle.
2) Evaluate
   a) It can run with high speed and accurately arrive destination.
3) Results
   As the attempt, we try to use MPC model in a condition which has no obstacle and make it to reach the destination. Although the result of algorithm is correct, but in simulation we find the robot will have obvious pause between each steps. The reason of stoppages is because the optimal process apply thousand times iteration to find the optimal value. But as consider the online time limitation, we abandon MPC models although the value return is optimal.

B. PID Controller

We did some experiment with different environments and models to evaluate our algorithm.

1) Experiment Fig 7
   a) Test Jackal robot in simulation environment with static obstacle.
   b) Test Jackal robot with sparse and dense obstacle environment.
2) Evaluate
   a) It could be reach destination and avoid all obstacles.

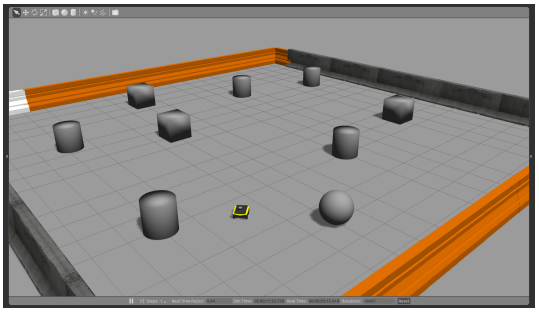

Fig. 7: Testing environment and static obstacles

b) It can run with high speed

3) Testing environment Following is our testing environment and the static obstacle we set. And when we are testing dynamic obstacles case, there are several people walking or running to test if Jackal can avoid.
4) Results
We have implemented the MPC Model and Solver with C++ code and run correctly as expected, but we are not satisfied with operating speed. In response to this question, what we use now is PID method. And we have great practical results.
We set two operation mode as run randomly and go to certain destination with both avoiding static and dynamical obstacles. And Jackal can smoothly avoid obstacles in both cases, and go to the expected destination in the latter case.
And the experimental Jackal speed is 1.8 meters per second, it can drive nearly full speed all the time in a rectangle condition with static and dynamic people walking and running without crashing.

Fig. 8: Testing environment and static obstacles

C. Human Tracking and Speed Estimate

In human tracking task, we need evaluate it without planning. Since the project has changed to use the **PID** control method which does not need pedestrians detection and tracking, we evaluate this system in a simulation environment (**ROS + Rviz**).

1) Evaluation methods
   a) Detecting and tracking human with a static robot.
   b) Detecting and tracking human with a dynamic robot.
2) Results
We recorded a few 30-second rosbag and hand label the pointcloud of people legs. And to evaluate the result, we define a detection/tracking is lost if the detection result and tracking result has changed its ID or lost in the Rviz for at least 3 seconds. And the average tracks are shown in the table below.

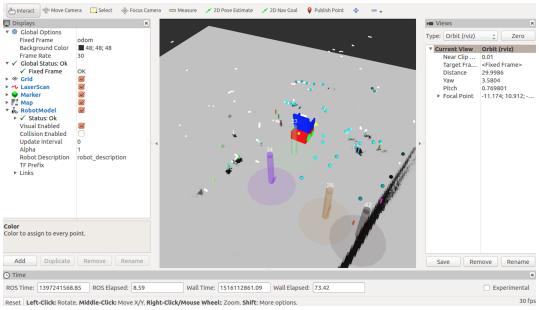

Fig. 9: One of the visulization results on Rviz.

TABLE I: Complexity of Conflict-free Coloring

Environment	Avg. duration	People tracked
Static environment	30.15s	23
Dynamic environment	29.87s	18

V. Conclusions

In this report, we introduce some related work and open-source software of this project. We talk about the detail of some algorithms which could be used in this project and discuss our work until now.

Till now, we has a PID framework which use practically great.

And we could detect people using our laser scanner, but it has many problems.

References

[1] R. G. Gosine, J. K. Sagawa, and J. Pannek, "Occupancy grid based distributed mpc for mobile robots."

[2] M. F. Manzoor, Q. Wu, and R. J. Masood, "Coordination control of wheeled mobile robot using mpc," in Computational Intelligence, Communication Systems and Networks (CICSyN), 2015 7th International Conference on. IEEE, 2015, pp. 241–246.

[3] D. Ferguson, M. Darms, C. Urmson, and S. Kolski, "Detection, prediction, and avoidance of dynamic obstacles in urban environments," in Intelligent Vehicles Symposium, 2008 IEEE. IEEE, 2008, pp. 1149–1154.

[4] B. D. Luders, S. Karaman, and J. P. How, "Robust sampling-based motion planning with asymptotic optimality guarantees," in AIAA Guidance, Navigation, and Control Conference (GNC), Boston, MA, 2013.

[5] A. Hussein, H. Mostafa, M. Badrel-din, O. Sultan, and A. Khamis, "Metaheuristic optimization approach to mobile robot path planning," in Engineering and Technology (ICET), 2012 International Conference on. IEEE, 2012, pp. 1–6.

[6] A. Leigh, J. Pineau, N. Olmedo, and H. Zhang, "Person tracking and following with 2d laser scanners," in Robotics and Automation (ICRA), 2015 IEEE International Conference on. IEEE, 2015, pp. 726–733.

[7] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5–32, 2001.

[8] "Jackal Simulator," http://wiki.ros.org/jackal_simulator/.

[9] "Veloyne Simulator," http://wiki.ros.org/velodyne_simulator/.

[10] "pointcloud to laserscan," http://wiki.ros.org/pointcloud_to_laserscan/.