neonavigation meta-package: 2-D/3-DOF Seamless Global-Local Planner for ROS –Development and Field Test on the Representative Offshore Oil Plant–

Atsushi Watanabe[1], Daisuke Endo[1], Genki Yamauchi[1] and Keiji Nagatani[1]

***Abstract*— This paper introduces an algorithm and implementation of a new navigation package for ROS. The contribution of this work is to provide a path and motion planner package that is suitable for the complex and narrow environments. The package realizes a 2-D/3-DOF seamless global-local planner. In this package, collision avoidance frequency can be faster than the conventional one, and planned path is always taking the global goal into account. The new navigation package is released as an open-source software on GitHub. In this paper, the field test results in the representative offshore oil plant, which has narrow corridors with projecting pieces of equipment, are shown.**

I. Introduction

The standard ROS (Robot Operating System) navigation meta-package was developed by Willow Garage and maintained by some ROS core members. It is reported that the navigation meta-package is suitable for long-distance navigation in an indoor office environment [1]. Also, it is known to be available in some simple outdoor environments.

Path-planning algorithms used in the conventional navigation meta-package are mainly Dijkstra search for the global planner and Dynamic Window Approach [2] and Trajectory Rollout [3] for the local planner. A layered 2D costmap framework is provided to combine static and dynamic maps.

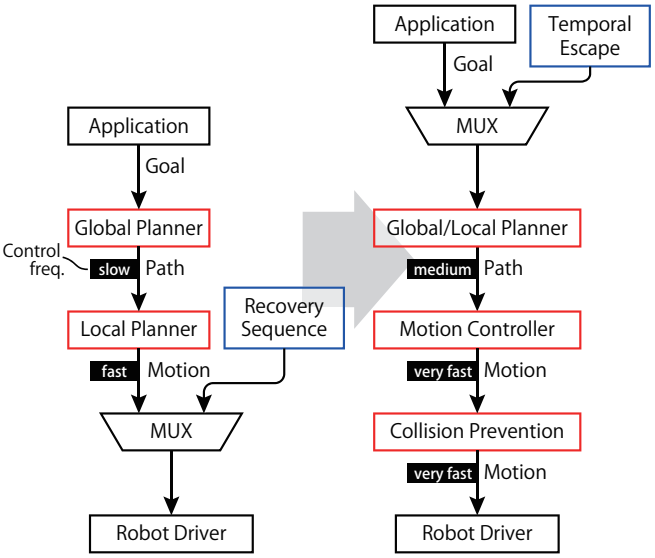
However, in the complex and narrow environments, for example, in industrial plants and messy indoor offices, several problems exist in the navigation provided by the ROS navigation meta-package.

- Separated global and local planners
  - Obstacle avoidance considers the only local area.
  - Global and local paths contradict and oscillate.
- Blind recovery behaviors
  - Recovery behavior plugins cannot use costmap data and planner functions.
  - Officially provided recovery behaviors are ad-hoc and cause physical crashes in complex environments.

The authors are developing a new 2-D/3-DOF seamless global-local planner ROS package by radically improving the above problems. The package is now an open-source software, distributed on GitHub: `https://github.com/at-wat/neonavigation/`.

On the other hand, the authors are developing a fully-automated inspection robot system as competition for the autonomous robot system for offshore gas and oil plants, called ARGOS (Autonomous Robot for Gas and Oil Sites) Challenge [4]. In this competition, small plants structured by temporary scaffold are used as a test field. The test field is made by an integrated oil company, Total, and it has almost the same equipment as the original one.

[1]All of the authors are with Field Robotics Laboratory, Tohoku University, New Industry Creation Hatchery Center, Aramaki-Aoba 6-6-10, Aoba-ku, Sendai, 980-8579, Japan, `atsushi.w at ieee.org`



a) conventional navigation package b) neonavigation package

Fig. 1. System structures of the conventional navigation meta-package and the new navigation meta-package, "neonavigation".

This paper introduces an algorithm and implementation of the new navigation meta-package and an example of the application for the ARGOS challenge. Also, field test results on the representative oil plant on the ARGOS challenge are shown.

II. Algorithms and Strategy

Figure 1 shows the difference of the standard system structures between the conventional navigation meta-package, and the new navigation meta-package named "neonavigation".

A. Conventional navigation meta-package

In the conventional navigation meta-package, global planner and local planner are separated and have a different control frequency. Typically, global planner frequency is above 1 Hz, and local planner is around 10 Hz.

In this structure, provided by the move_base package, local planner simultaneously does motion planning considering kinematics of the robot hardware and obstacle avoidance. Since the local planner has the role of obstacle avoidance,

its control frequency must be fast enough according to the speed of the robot.

Also, since the local planner does not have the global goal information, the planner sometimes selects the local minimum in the local costmap or oscillated motion.

Moreover, since the recovery behaviors are completely independent of the planners and overwrite the planner result, most of the recovery behaviors provided in the navigation meta-package are the motion sequence without any sensor information; recovery behaviors cause collisions in complex and narrow environments.

B. neonavigation meta-package

In the neonavigation meta-package, the motion control that controls the vehicle to follow the generated path based on [5] and collision prevention are separated from the planner; then the motion control and collision prevention can have the faster control frequency. For example, global/local planner frequency is around 5 Hz, and the motion control and collision prevention can be 50 Hz.

Also, the global and local path are seamlessly and simultaneously planned based on the A* algorithm. The global plan is included as a heuristic function of the A* as a distance from the goal. The distance from the goal can be preliminarily calculated when the goal is given and the environment is changed. Therefore, the A* search result from the start with censoring at the necessary distance for the local plan can always be a globally optimum path in the measurable world.

In the neonavigation meta-package, the recovery behavior temporarily overwrites the goal of the application; collision avoidance and prevention are always available.

III. NEONAVIGATION META-PACKAGE

The neonavigation meta-package currently has five packages: costmap, planner, safety_limiter, trajectory_tracker, and neonavigation_launch packages. The neonavigation_launch package contains an example launch file for testing and a skeleton to be used in the application.

Figure 2 shows the system structure of the navigation system and data flow provided by the neonavigation meta-package. Mainly the following four nodes in these packages are used in the autonomous navigation. Currently, these nodes do not have nodelet plugins; it remains as a future work.

A. costmap_3d

The costmap_3d node generates and updates 2-D/3-DOF costmap from 2-D maps. This node subscribes global static 2-D map and local dynamic 2-D map. The generated costmap is an occupancy gridmap in x, y, and yaw configuration space.

Figure 3 shows an example of the generated 2-D/3-DOF costmap where the z-axis represents yaw orientation of the robot. Occupancy of the grid near the obstacle is calculated by the footprint and orientation of the robot.

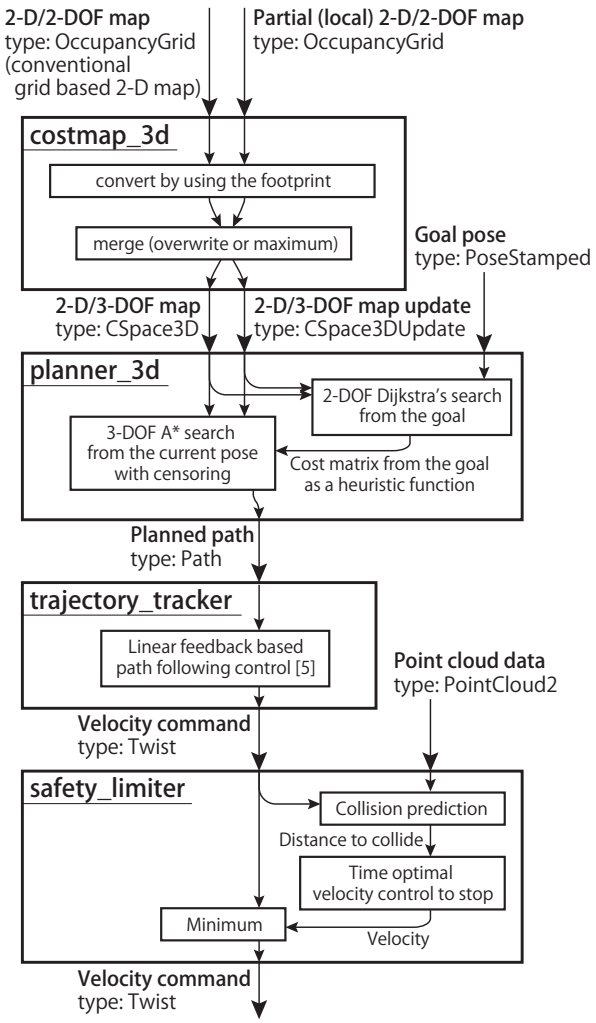


Fig. 2. Data flow diagram of the navigation using neonavigation meta-package.

B. planner_3d

The planner_3d node is based on the A* algorithm with the heuristic function of the distance from the goal considering obstacles. The distance from the goal is calculated and held as a distance map and is partially updated triggered by the update of the costmap. This works as a global planner.

The state transition is searched in the configured range on the x-y surface and every orientation. Also, the kinematic constraint cuts branches of the search to decrease computation cost, as shown in Figure 4. Only states that are connectable from the current state by a circular arc are alive in the search. The search from the current robot pose to the goal can be censored at the configured distance. Still, the search result is a sub-optimal local path since the global distance from the goal is always considered.

Also, to reduce the oscillation and the computation cost, distance from the planned path in the previous cycle is added to the path search cost function. This works as a hysteresis of the path selection.

For example, as shown in Figure 5, paths with switch-back

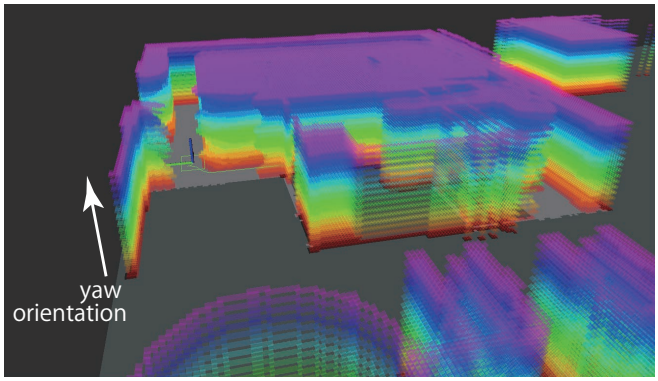


Fig. 3. Visualized 3-DOF costmap in (x, y, yaw) configuration space.

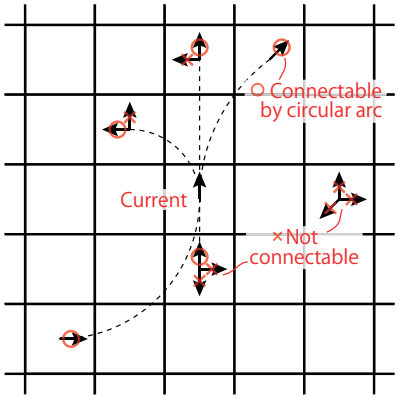


Fig. 4. Grid search with kinematic constraints.

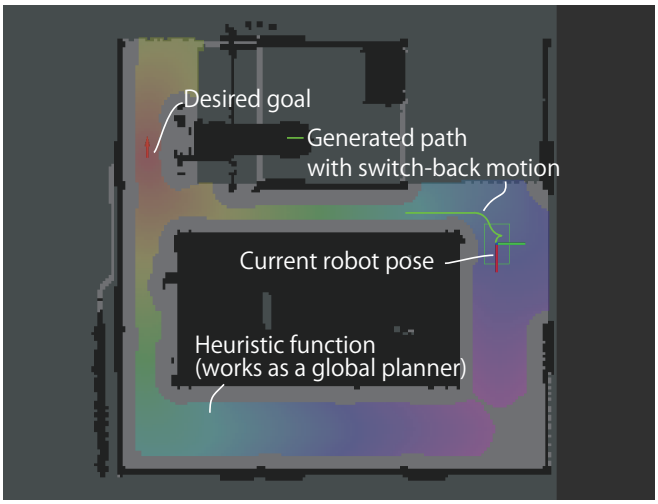


Fig. 5. Grid search with kinematic constraints.

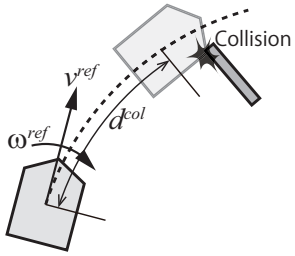


Fig. 6. Controlled variables of the collision prevention.

motion can be planned. Grid search algorithm is optimized by considering the processorfs cache and re-ordering near grids to have closer addresses. As a result, this planner can work in around 5 Hz on the Core i7 processor at an average of less than 50

In some cases, with a complex narrow place in that difficult-to-find path, the planner gets slower compared to the conventional navigation meta-package since it considers the relatively larger number of possible motions. In this case, a watchdog timer in the planner publishes zero velocity commands until the path is planned. After finding the path in such a case, the planner gets faster since the path near the previous one is preferentially searched due to the hysteresis factor of the cost function.

For recovery from a stuck position, the goal pose is temporarily overwritten by the random pose around the current pose if no available path is found. This makes the movement of the point of view of the sensor and helps it to escape from being stuck. During this, path planning and collision prevention are always enabled.

C. *safety_limiter*

The safety_limiter node prevents collision to the obstacle by decreasing the linear and angular velocity. This node predicts the pose of the robot by linear extrapolation according to the current velocity command, and determines the distance to the collision, as shown in Figure 6. The velocity is decreased based on the following expressions and are time optimal controls to stop d^{mergin} before the collision.

$$v^{lim} = \mathrm{sgn}\,(d^{col} - d^{mergin})\,\mathrm{sgn}\,(v^{ref})\sqrt{2\alpha\,|d^{col} - d^{mergin}|}$$
$$\omega^{lim} = \omega\frac{v^{lim}}{v^{ref}} \tag{1}$$

where α is a desired acceleration of the robot motion.

Input and output velocity command of the safety_limiter always have the same turning diameter; the resultant path of the robot does not change from the given path even if the safety_limiter is used.

D. *trajectory_tracker*

The trajectory_tracker controls the vehicle's linear and angular velocity, to follow the given path based on [5]. This implementation was reported in [6].

The control is the linear feedback of the distance, angle, and angular velocity error against the given path to the robot's angular acceleration. These controlled variables are defined, as shown in Figure 7.

IV. Field Test on the Representative Offshore Oil Plant

The developed neonavigation meta-package was tested in the ARGOS Challenge competition held by Total.

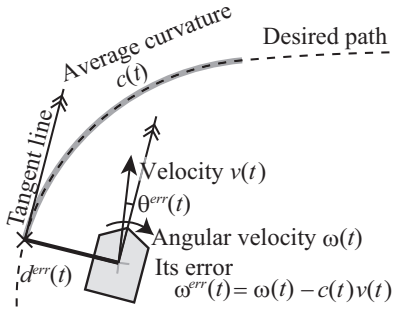


Fig. 7. Controlled variables of the path following control based on [5].

The site, named "UMAD," is the representative offshore oil plant in Lacq, France that imitates a typical Total petroleum processing environment. The following tests were done for the second of three times in the competitions held on April 4 to 8, 2016.

A. Scenario of the test

Figure 8 shows the picture of the UMAD. At the time of the competition, the ground, first, and second floor were used in missions.

The robot system is required to navigate through several floors and inspect pressure gauges, valve positions, water level gauges, and temperature of the equipment. The operation of the system must be fully automated and ask human operator permission or selection of the behavior only in abnormal situations. In some of the missions, network disconnections, alert sirens, ultrasounds of gas leaks, and high-temperature sources appear. After recognizing these abnormal states, the robot has to go to the safe area, find the position of the abnormal source, and report it to the operator.

Table I shows summaries of the given missions. The missions were given as a text with a sketch of the positions, and the authors made mission scripts that contain positions to visit and measure.

B. Autonomous mobile robot system using neonavigation meta-package

Figure 9 shows the robot system, "AIR-K 1.5", used for the test in the ARGOS challenge. The AIR-K 1.5 is based on the 6-crawler mobile robot "kenaf" [7] that has two tracks and four sub-tracks.

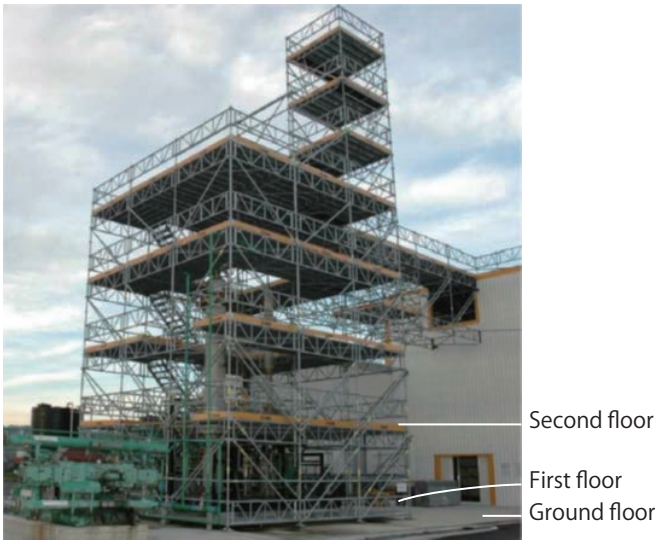


Fig. 8. The representative offshore oil plant "UMAD" in the ARGOS (Autonomous Robot for Gas and Oil Sites) Challenge competition held by Total.

The AIR-K 1.5 has two 3-D LIDARs, Hokuyo YVT-X002 [8] and an IMU for the autonomous navigation, four cameras for the measurement of the gauges, a thermal camera for detecting heat sources, and an ultrasound microphone for detecting gas leak sounds. The robot is connected to the operation room through wireless LAN to report the status of the robot system and receive mission updates.

Figure 10 shows a system software structure of the AIR-K 1.5. The goal given to the planner is applied by mission_manager. The system has two other motion planners: "step_traverse" for climbing and descending the steps and "cv_measure" for changing the posture and measuring the gauges. The output of the motion planner is multiplexed according to the mission_managerfs decision. Pointcloud data from 3-D LIDAR is provided by "hokuyo3d" node, and the occupancy gridmap for the navigation is generated by "travelable_area" node. Also, the static map of the each floor is selected by "select_map" node.

The AIR-K 1.5 uses an implementation of the Monte Carlo localization, "amcl" package, from the conventional navigation meta-package. The localization is performed in the 2-D map of each floor using a clipped from 3-D LIDAR

TABLE I

SUMMARY OF THE MISSIONS

mission	start	goal	points to be inspected	special
#1	Starting area	Starting area	Measure the pressure gauge CP1, and the valve CP14 on the second floor	
#3	Safe area	Starting area	Visit VP1, VP2, and VP3	An unknown obstacle exists in the UMAD
#4	Starting area	Starting area	Look around and find heat source	
#5	Starting area	Starting area	Measure the valve CP5 and the pressure gauge CP7, CP1, and CP3	
#6	Starting area	Starting area	Measure the valve CP4 and the water level gauge CP9	Several abnormal state and additional mission will be given during the mission

* #2 was a free mission to be defined by each participent.

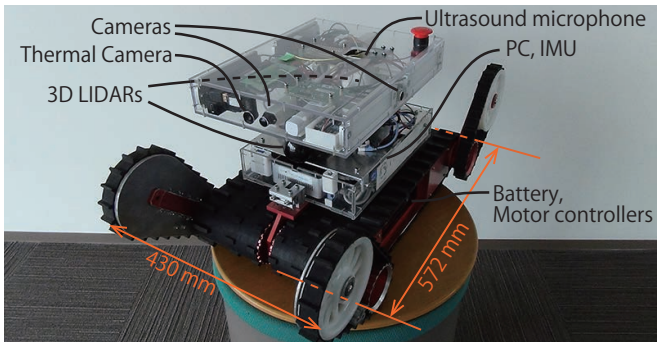


Fig. 9. The robot system, called AIR-K 1.5, for the test in the ARGOS challange, based on the 6-crawler mobile robot "kenaf" [7]

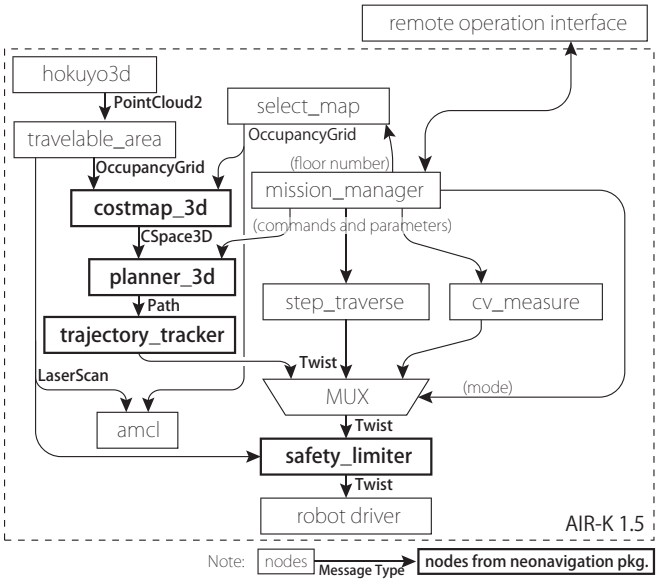


Fig. 10. Software system structure of the AIR-K 1.5 excepting measurement system

data.

C. Tests and results

The #1, #3, #4, #5, and #6 missions were performed in the test. Figure 11 shows a picture of the robot system on UMAD. The floor surface has lugs and holes, and the robot gets large disturbance.

Figure 12–14 show examples of the navigation results of the missions. The navigation through narrow corridors in these missions is successfully done and never crashes into the obstacles, even in the case of localization failure.

Results of the 9 trials of the five missions are shown below. Problems related to the neonavigation meta-package are italicized in the lists.

mission #1 try-1 (aborted)

1) localization failure (**retried from the start**)

mission #1 try-2 (completed)

mission #3 (completed)

1) *vibrative path during passing through very narrow place (**autonomously solved**)*

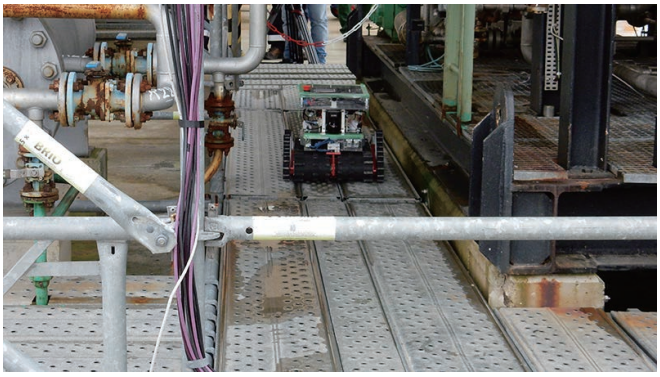


Fig. 11. Picture of the robot on UMAD

2) *soft collision to the environment (**continued**)*
3) localization failure (**corrected position and continued**)

mission #4 try-1 (completed)

mission #5 try-1 (aborted)

1) stack after climbing the step due to a bug in mission_manager (**retried from the start**)

mission #5 try-2 (completed)

1) sub-track hardware problem (**continued**)

mission #4 try-2 (aborted)

1) sub-track hardware problem (**retried from the start**)

mission #4 try-3 (completed)

1) *soft collision to the environment (**continued**)*

mission #6 (aborted)

1) localization failure (**corrected position and continued**)
2) sub-track hardware problem (**continued**)
3) stair descending motion failed due to a bug (**mission stopped**)

V. Lessons Learned and Future Works

It is confirmed that the robot system with the neonavigation meta-package is suitable for navigating a plant with a narrow corridor. However, quantitative comparison to the conventional navigation meta-package remains for future work.

Fatal problems on the tests were mainly caused by localization and mechanical problems. Since this environment has sparse wall structure by the temporary scaffold, 2-D localization by using clipped 3-D pointcloud is not enough. 3-D localization, with the constraint of the ground vehicle, might improve this problem.

On the other hand, it is clear that the minor problems still exist on the neonavigation meta-package. In some cases, the generated path was vibrative, especially while passing through the very narrow place. This problem seems to be caused by the difference in the viewpoints of sensing the obstacle and planner cycle. However, the paths were finally converged autonomously.

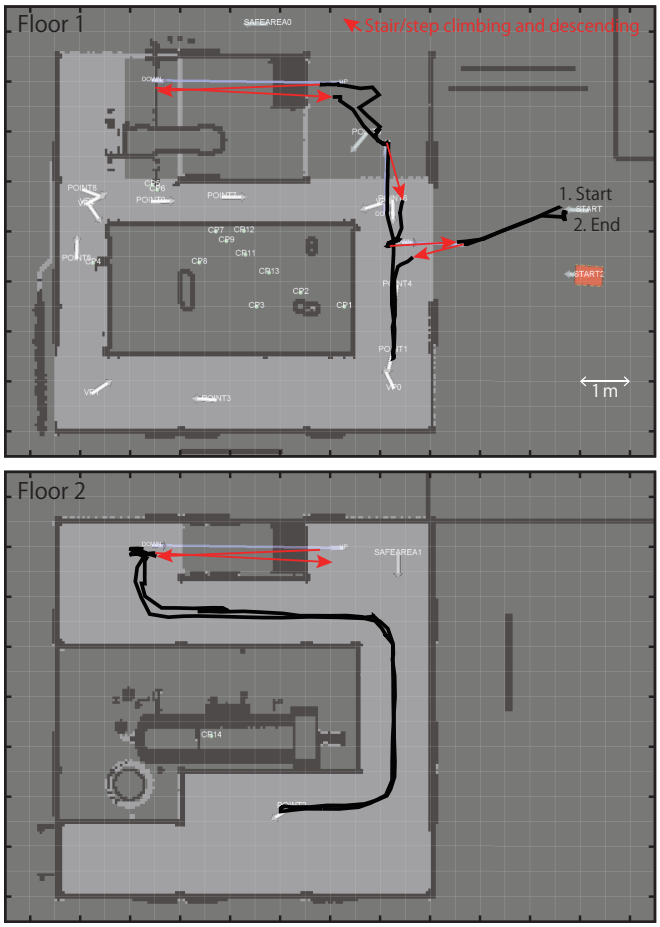


Fig. 12. Navigation result of the mission #1 (try-2)

Fig. 13. Navigation result of the mission #4 (try-3)

Also, soft collisions to the environment occurred. This problem seems to be caused by the lack of sensing ability, and hardiness to move as decided due to the rough floor surface.

Improvement of the planner to decrease vibrative motion, improvement of the path-following controller and collision prevention remain for future works. Also, a quantitative evaluation of the package and tests on more of the environments are required.

Fig. 14. Navigation result of the mission #5 (try-2)

VI. Conclusion

This paper introduced an algorithm and implementation of the new navigation meta-package for ROS. The package realizes a 2-D/3-DOF seamless global-local planner. This work provides a path and motion planner package for ROS that is available in the complex and narrow environments. The main advantage of this meta-package is that the collision avoidance frequency can be faster than the conventional, planned path, always taking the global goal into account.

The ability of the developed new navigation meta-package was confirmed by the field tests in the representative offshore oil plant with narrow corridors. The authors are planning to perform a quantitative comparison to the conventional navigation meta-package.

References

[1] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *International Conference on Robotics and Automation*, 2010.

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

[3] B. P. Gerkey and K. Konolige, "Planning and control in unstructured terrain," in *In Workshop on Path Planning on Costmaps, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA*, 2008.

[4] K. Kydd, S. Macrez, P. Pourcel *et al.*, "Autonomous robot for gas and oil sites," in *SPE Offshore Europe Conference and Exhibition*. Society of Petroleum Engineers, 2015.

[5] S. Iida and S. Yuta, "Vehicle command system and trajectory control for autonomous mobile robots," in *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, 1991, pp. 212–217 vol.1.

[6] Y. Morales, A. Watanabe, F. Ferreri, J. Even, T. Ikeda, K. Shinozawa, T. Miyashita, and N. Hagita, "Including human factors for planning comfortable paths," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6153–6159.

[7] T. Yoshida, E. Koyanagi, S. Tadokoro, K. Yoshida, K. Nagatani, K. Ohno, T. Tsubouchi, S. Maeyama, I. Noda, O. Takizawa *et al.*, "A high mobility 6-crawler mobile robotfkenaff," in *Proc. 4th International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster (SRMED2007)*, 2007, p. 38.

[8] K. Kimoto, N. Asada, T. Mori, Y. Hara, A. Ohya *et al.*, "Development of small size 3d lidar," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 4620–4626.