Simultaneous Adaptive Path Planning System for Planetary Rover

Yoji KURODA, Daisuke ARITA, Teppei SAITO
Meiji University
{ykuroda, ce66204, ce66232}@isc.meiji.ac.jp

Abstract

In this paper, we propose a path planning system named SAP (Simultaneous Adaptive Path planning) which can make a plan adapt to kinematic and dynamic constraints, and dynamically changing environment simultaneously. SAP has three key issues: condition layers, a dynamic space, and a search tree. Each condition layer involves a factor which influences running of the robot such as coefficient of friction and elevation of the field. Dynamic space is an area in velocity domain where the robot could reach in the next time step. The limitation (means outer shape) of the dynamic space is defined as a function of all conditions and a robot model, velocities that can be achieved within a short time interval. Search tree that consider the dynamic space expands like tree to search a path from start to goal. By these mechanisms, SAP can generate a path that robot can achieve in real environment.

1. Introduction

Path planning is one of the key issues which make a planetary surface explorer (rover) accomplish a mission. In order to plan a successful path where the rover could track, we should consider various problems as follows. First, the generated path must be considered whether the rover could exist in safe. Not only the collision with obstacles but also potential risk of dropping off into a ditch, because the surface of planet is composed of complex topography.

Second, the rover should explore planetary surface even if the rover is blocked a passage by steep slope and slippery field, etc. Rover should run through these fields by considering of the kinematics and the dynamics of the rover.

Third, there might be many moving obstacles on planetary surface. The rover has to have the ray of sun to generate energy with solar paddles. But the positions of shadows change according to the surface topography and almanac. Greater surface topography could be known a prior from the orbit, but very small shadow which can not be seen from there may be big enough for the rover. Therefore, shadows could be regarded as moving obstacles.

Fourth, limitations of time are critical factors. The time of the exploration may be strictly restricted by many reasons; therefore the planning of rover should estimate the time.

To solve the above-mentioned problems, path planner should adapt to kinematic and dynamic constraints, and dynamically changing environment simultaneously. However, though quite a lot of effective algorithms have been studied, they are not enough yet.

In early years, there are three general approaches of path planning: roadmap, cell decomposition, and potential field. The roadmap approach captures the connectivity of collision-free space in a network of one-dimensional curves, called the roadmap. After constructing roadmap, we can find the shortest path between the initial and goal configuration by applying search algorithm such as Dijkstra’s algorithm. The configuration of a robot is a set of parameters that determine the position and attitude of the robot. Classic examples of the roadmap approach are visibility graph algorithm and voronoi diagram algorithm. Visibility graph is built by connecting the initial and goal configuration with edges of all obstacles in the given map [1]. Voronoi diagram leads through the middle of available corridors between obstacles [2].

Cell decomposition method divides the robot’s free space into several regions, so called cells. We construct a graph to capture the connectivity of collision-free space, just as the roadmap approach do. The nodes of a graph are the cells. There is an edge between two nodes if the corresponding cells are adjacent to each other. We can find the shortest path between the initial and goal configuration by applying search algorithm [3] [4].

Potential Field approach appears of a somewhat different nature from the previous two. The approach does not build a connectivity graph. Instead, the
method constructs artificial potentials field. Different potentials are assigned to the cells of the grid. “Attractive” potentials are given to cells close to the robot’s goal. “Repulsive” potentials are assigned to obstacles. A path is constructed along the most promising direction [5].

These three general approaches are basic of a lot of path planning algorithms. These three general approaches are very useful as far as a free space has already been set and drawing the line from start to goal. However, these approaches do not consider kinematics and dynamics. For example, when the path is drawn abeam, automobile can not trace the path. Then, the approach that adapt to kinematics and dynamics has been proposed in path planning.

There is nonholonomic planner as an algorithm that specializes in kinematics of a robot. In a nonholonomic planner, a path is created as a set of maneuvers, which take into account kinematic constraints of a robot [6]. KDP (Kinodynamic Planning) is a path planning method that considers kinematic and dynamic constraints. In [7], [8], randomized kinodynamic approach is presented to planning a path in a static environment. The state space of a robot, e.g., position, velocity, is explored by applying a set of allowable control inputs in order to grow a tree. The exploring of the state space is complete when a creating branch is reached to the goal configuration. As an example of dynamic environments, kinodynamic planning among moving obstacles was proposed [9].

KDP is effective algorithm that considers the kinematics and dynamics of the robot. However, KDP is not enough in real environment, because KDP do not consider the running environment of the robot such as slope and slippery field. To discuss the effective path planning in real environment, it is necessary to consider not only the field of path planning but also the field of navigation of robot.

Generally, navigation of robot has the two methods of model-based and sensor-based. Path planners shown above are model-based approach. The research of path planning and control are classified in model-based approach. Generally, a robot achieves an autonomous movement by controlling the robot to track the path generated with path planning. In sensor-based approach, the robot without information on the environment plans the next action based on the sensor data. The robot reaches the goal while avoiding the obstacle.

Dynamic window approach is one of sensor-based approach [10]. The dynamic window approach is an obstacle avoidance method that takes into account the kinematic and dynamic constraints of a robot. Kinematics and dynamics are taken into account by directly searching the velocity space of a robot. The search space is the set of parameters (v, w) of translational velocities v and rotational velocities w that are achievable by the robot. Sensor-based approach considers dynamics and kinematics of the robot more than model-based approach. However, the robot might not reach the goal because of planning only next action.

In [11], [12], it is described that a motion control method for mobile robots based on integration of model-based approach and sensor-based approach. The motion control method utilizes the advantage of model-based approach and sensor-based approach. The method prevents the robot from deviating from the path by a theoretical calculation. Therefore, the motion control method is not discussed that the robot deviates from the global path. However, in real world, it is difficult for the robot to follow the path completely. The problem is important in practical use. Generally, there are two methods when the robot deviates from the path. One is a method of the return to former path [13]. Another is a method of the replanning. The former has a problem that robot tries to return to previous path even if a better path is found newly. The latter also has a problem of taking time when planning again.

Common problem in above-mentioned approaches is to describe the area that can not pass only as obstacle. As discussed previously, when the path is drawn abeam, automobile can not trace the path. Moreover, some possible controls are limited further on the steep slope and the slippery field, etc. These examples show that path planner should consider not only the area that robot can pass and obstacle but also factor that influence the kinematics and the dynamics of the robot in running environment.

In this paper, we propose a path planning system named SAP (Simultaneous Adaptive Path planning) which can make a plan adapt to kinematic and dynamic constraints, and dynamically changing environment simultaneously. SAP consider not only the area that robot can pass and obstacle but also factor that influence the kinematics and the dynamics of the robot in running environment. Moreover, SAP can generate a path that considers robot states, e.g., a set of velocities, positions, postures, and time. Therefore, SAP enables the robot to avoid the moving obstacle and take advantage of the kinematics and the dynamics of the robot. As the result, SAP can generate the path which adapts to kinematics, dynamics, and dynamically changing environment simultaneously. SAP
successfully accomplishes autonomous navigation in numerical simulations and real world experiments.

2. Simultaneous Adaptive Path Planning

Simultaneous adaptive path planning (SAP) is a path planning system that can adapt to kinematic and dynamic constraints, and dynamically changing environment simultaneously. SAP has three key issues: condition layers, a dynamic space, and a search tree.

We define “condition” as a factor which influences running of the robot. SAP has layered architecture as shown in Fig. 1. Various conditions are divided into layers of each condition. We define the layer of each condition “condition layer” in SAP. The condition layer is composed of the coordinate grid. Each grid has the value which shows the running environment of the robot such as coefficient of friction and elevation of the field.

Dynamic space is an area in velocity domain where the robot could reach in the next time step. The limitation (means outer shape) of the dynamic space is defined as a function of all conditions and a robot model, velocities that can be achieved within a short time interval. Some of combinations of translational and rotational velocities might be chosen probabilistically in dynamic space, and integrated. As results of some integration in a short time span, some new points where show the expected states of the robot – defined as nodes - are placed on a planning map. Each node involves robot’s status, e.g., a set of velocities, positions, postures, and time. By repeating the operations until reaching the goal point, paths that various conditions are considered are generated as a tree. We define the tree as a search tree.

By these mechanisms, SAP can generate the path which adapts to kinematics, dynamics, and dynamically changing environment simultaneously. We describe SAP more in detail as follows.

2.1. Procedure

2.1.1. Condition layer. SAP includes not only the area that robot can pass and obstacle but also condition layer that influences the kinematics and the dynamics of the robot in running environment.

Fig. 1 shows the architecture of the condition layers. The condition layer is composed of the coordinate grid. Each grid has the value which shows the running environment of the robot such as coefficient of friction and elevation of the field. Given the position of the robot, SAP uses the value at the position from condition layer.

2.1.2. Dynamic space. The force to move the robot is limited by performance of the robot (e.g., power, torque) and those of conditions. Dynamic space is an area in velocity domain where the robot could reach in the next time step. The limitation (means outer shape) of the dynamic space is defined as a function of all conditions and a robot model, velocities that can be achieved within a short time interval. Some of combinations of translational and rotational velocities might be chosen probabilistically in dynamic space.

2.1.3. Generate a new node. Some new points are generated by the results of integrations with selected translational and rotational velocities. We defined the new point as node that involves robot’s status, e.g., a set of velocities, positions, postures, and time.
2.1.4. **Search tree.** By selecting stochastically from some nodes and repeating the procedure 2.1.1-2.1.3, paths extend like a tree. We defined the tree as search tree that shows Fig.4.

![Fig. 4 Searching an admissible path](image)

2.1.5. **Global path.** A global path is selected from search tree, when the node reaches goal point. When global path is not found in the time limit, SAP would return the best effort result or plan a new path.

2.1.6. **Control and replanning.** Global path has the information on velocities, positions, postures, and time. The robot is operated based on global path. When the robot deviates from the path while the robot is being operated, path is planned again. Because the processing time is short, SAP can plan the global path again in real time despite a lot of information.

2.2. **Definition**

2.2.1. **Kinematics and Dynamics.** Kinematics is a branch of mechanics that describes the motion of objects without the consideration of the masses or forces that bring about the motion. In contrast, dynamics is concerned with the forces and interactions that produce or affect the motion.

In real environment, a variety of kinematic and dynamic constraints exist. If a robot does not satisfy kinematic and dynamic constraints even if not colliding with the obstacle, the robot cannot achieve the state. In order to move a mobile robot in real world, path planner has to generate at least an admissible path under these constraints.

The kinematic and dynamic constraints are affected from not only the characteristics of the robot but also the environment. For example, in an automobile, if you turn the steering wheel all the way on icy road, the automobile may slip out. This is because the environment of the icy road restricts admissible state more. There are a lot of factors that cause the kinematic and dynamic constraints such as slipperiness and elevation. SAP divides those factors into each condition layer.

2.2.2. **Obstacles.** We define an obstacle as a movable space that the robot cannot pass through obviously or a movable space that the robot should not pass through. SAP has a predictor that calculates the velocity vectors of the obstacles. If the vectors were known, SAP could generate a path that avoids obstacles easily. For example, in the case of the robot having solar paddles, shadow could be regarded as the obstacles disturbing the ray of sun. Predictor calculates a velocity vector of the shadow with sun angle and an altitude. As a result, SAP can generate a path that avoids a shadow.

3. **Simulation and Experiment**

3.1. **Robot Model**

In this paper, we use different model in the simulation and the experiment to show that SAP can adapt to a variety of robot models. The robot model of simulation is the two wheels model, and the robot model of experiment is the skid-steer model.

Travel resistance and friction affect motion of the vehicle. We calculate the dynamic space from travel resistance and friction that hangs to each tire as follows. Fig. 5 shows two wheels model, and Table I shows nomenclature of two wheels model. Fig. 6 shows skid-steer model, and Table II shows nomenclature of skid-steer model.

3.1.1. **Travel resistance.** Travel resistance is one of conditions. Travel resistance is a force to prevent running of the vehicle. If travel resistance is larger than drive power, the vehicle could not be accelerated. The travel resistance mainly consists of rolling resistance, air resistance, grade resistance, and accelerating resistance.

A) **Rolling resistance**

When the tire is rolling on the road, the rolling resistance is generated. It is shown by the following equation.

\[ R_r = C_{rr} \cdot mg \cdot \cos(\text{pitch}) \]  \hspace{1cm} (1)

where,

- \( R_r \): Rolling resistance [N]
- \( C_{rr} \): Rolling resistance coefficient
- \( m \): Vehicle weight [kg]
- \( g \): Acceleration of gravity [m/s^2]
- \( \text{Pitch} \): Pitch angle [rad]
B) Air resistance

When the vehicle is running, front air pressure is higher than back pressure. As a result, a force that prevents running of the vehicle is generated. The air resistance is shown by the following equation.

\[ R_d = \frac{\frac{\rho AV^2}{2}C_d}{2} \]  

\[ \text{where,} \]

\[ R_d: \text{Air resistance [N]} \]
\[ \rho: \text{Air density [kg/m}^3\text{]} \]
\[ A: \text{Frontal area [m}^2\text{]} \]
\[ V: \text{Vehicle speed [m/s]} \]
\[ C_d: \text{Air resistance coefficient} \]

C) Grade resistance

The grade resistance is generated when vehicle climbs a slope. Grade resistance is a decelerating force factor at the uphill slope. Conversely, grade resistance is an accelerating force factor at the downhill slope. The grade resistance is shown by the following equation.

\[ R_u = \frac{m g}{2} \sin(\text{pitch}) \]  

\[ \text{where,} \]

\[ R_u: \text{Grade resistance [N]} \]
\[ m: \text{Vehicle weight [kg]} \]
\[ g: \text{Acceleration of gravity [m/s}^2\text{]} \]
\[ \text{pitch: Pitch angle [rad]} \]

D) Accelerating resistance

When the vehicle is accelerated, accelerating resistance is generated. The accelerating resistance is shown by the following expression.

\[ R_a = mg \alpha \]  

\[ \text{where,} \]

\[ R_a: \text{Grade resistance [N]} \]
\[ m: \text{Vehicle weight [kg]} \]
\[ g: \text{Acceleration of gravity [m/s}^2\text{]} \]
\[ \alpha : \text{Acceleration [m/s}^2\text{]} \]

3.1.2. Friction of each tire. Friction of each tire is calculated from vertical load of each tire calculated from the attitude of the vehicle and coefficient of friction of the road. The load shifts of the tire are calculated by equation (5) and (6).

\[ \Delta W_{\text{roll}} = \frac{mg \sin(\text{roll})}{T} H \]  

\[ \Delta W_{\text{pitch}} = \frac{mg \sin(\text{pitch})H}{l} \]

Friction \( F \) of each tire is calculated by the following equation.

\[ F_{1,2,3,4} = \mu W_{1,2,3,4} \]  

3.1.3. Drive power. Drive power is usually assumed to be constant. However, if the drive power exceeds the friction limit, SAP would choose the drive power within the friction limit. We use the vehicle of front engine, front-wheel drive in this paper. The limit speed of the vehicle is calculated from travel resistance and drive power by the following equation.

\[ V_{\text{limit}} = \sqrt{\frac{2 \times \frac{F_{\text{power}} - R_d - R_u}{\rho C_d A}}{2}} \]  

3.1.4. Trajectory of the two wheels model. Cornering force is the counter power of the centrifugal force, which is appeared on each tire when the robot is cornering. These forces work in the right-angled to the traveling direction of the vehicle. When the front wheel reaches the friction limit before the rear wheel does, the vehicle drifts out. When the rear wheel reaches the friction limit before the front wheel does, the vehicle spins out. We assume that maximum cornering force \( Y \) equal to friction \( F \).

\[ Y_{f,r} = F_{f,r} \]  

In addition, when we assumed that cornering power \( Y \) is proportional to the sideslip angle \( \beta \), the limit of sideslip angle is calculated by the following equation.

\[ \beta_{f,r} = \frac{Y_{f,r}}{K_{f,r}} \]  

\( K \) is cornering power. The following equations are calculated from Fig. 5.

\[ Y_f = -K_f \frac{\beta + l_f}{V - \delta} \]  

\[ Y_r = -K_r \frac{\beta - l_r}{V} \]  

\[ mVr = 2Y_f + 2Y_r \]  

The limit steering angle is the following equation from (10), (11), (12).

\[ \delta_{\text{lim}}, \alpha = \frac{2(Y_f + Y_r) + mg \sin(\text{roll})}{mV^2} l + \beta_f - \beta_r \]

Desired speed and yaw rate of a vehicle are decided within the dynamic space. Sideslip angle \( \beta \) and yaw rate \( r \) are calculated from \( V \) and \( \delta \). The equation is shown as follows.

\[ \left[ \begin{array}{c} 2\left(l_f, K_f, l_r, K_r, \beta_f, \beta_r \right) \end{array} \right] \frac{mV + 2\left(\beta_f - l_f, K_f, \beta_r - l_r, K_r, \beta_f \right)}{V} = \left[ \begin{array}{c} 2\left(l_f, K_f, l_r, K_r, \beta_f, \beta_r \right) \end{array} \right] \delta \]  

\[ \left[ \begin{array}{c} 2\left(l_f, K_f, l_r, K_r, \beta_f, \beta_r \right) \end{array} \right] \frac{mV + 2\left(\beta_f - l_f, K_f, \beta_r - l_r, K_r, \beta_f \right)}{V} \]
We assume the position of the center of gravity of a vehicle expressed in the global coordinates to be $X$ and $Y$.

\[
\begin{aligned}
\frac{dX}{dt} &= V \cos(\beta + \theta) \\
\frac{dY}{dt} &= V \sin(\beta + \theta) \\
\frac{d\theta}{dt} &= r
\end{aligned}
\]  

(16)

### Table I

**NOMENCLATURE OF TWO WHEELS MODEL**

<table>
<thead>
<tr>
<th>Parameters [Unit]</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle mass [kg]</td>
<td>$m$</td>
</tr>
<tr>
<td>Front tire cornering power [N/rad]</td>
<td>$K_f$</td>
</tr>
<tr>
<td>Rear tire cornering power [N/rad]</td>
<td>$K_r$</td>
</tr>
<tr>
<td>Wheel base [m]</td>
<td>$l$</td>
</tr>
<tr>
<td>Distance from center of gravity to front axle [m]</td>
<td>$l_f$</td>
</tr>
<tr>
<td>Distance from center of gravity to rear axle [m]</td>
<td>$l_r$</td>
</tr>
<tr>
<td>Yaw rate [rad/s]</td>
<td>$r$</td>
</tr>
<tr>
<td>Slip angle of vehicle [rad]</td>
<td>$\beta$</td>
</tr>
<tr>
<td>Slip angle of front tire [rad]</td>
<td>$\beta_f$</td>
</tr>
<tr>
<td>Slip angle of rear tire [rad]</td>
<td>$\beta_r$</td>
</tr>
<tr>
<td>Velocity [m/s]</td>
<td>$V$</td>
</tr>
<tr>
<td>Front tire steer angle [rad]</td>
<td>$\delta$</td>
</tr>
</tbody>
</table>

![Fig. 5 Two Wheels Model](image)

3.1.4. **Trajectory of the skid-steer model.** The wheel of skid-steer model is assumed not to slip in the experiment. Table II shows the symbols in the following equations (17), (18), (19) and (20). Speed and yaw rate ($v$, $w$) of the robot are the following equations when the control input is rotational speed of right and left wheel.

\[
\begin{pmatrix}
    v_R \\
    v_L
\end{pmatrix} =
\begin{pmatrix}
    r \phi_R \\
    r \phi_L
\end{pmatrix}
\]

(17)

\[
\begin{pmatrix}
    \dot{v} \\
    \dot{w}
\end{pmatrix} =
\begin{pmatrix}
    \frac{r}{2} & \frac{r}{2} \\
    \frac{-r}{2d} & \frac{-r}{2d}
\end{pmatrix}
\begin{pmatrix}
    \phi_R \\
    \phi_L
\end{pmatrix}
\]

(18)

The range of $v$ in dynamic space is calculated by expression (8). Therefore, the range of $w$ in dynamic space is decided within the range to satisfy the following expression.

\[
\frac{r \dot{\phi}_R + r \dot{\phi}_L}{2} < V_{\text{lim}, \theta}
\]

(19)

We assume the position of the center of gravity of a vehicle expressed in the global coordinates to be $X$ and $Y$.

\[
\begin{aligned}
\frac{dX}{dt} &= v \cos(\theta) \\
\frac{dY}{dt} &= v \sin(\theta) \\
\frac{d\theta}{dt} &= w
\end{aligned}
\]

(20)

### Table II

**NOMENCLATURE OF SKID-STEER MODEL**

<table>
<thead>
<tr>
<th>Parameters [Unit]</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance from center of gravity to wheel [m]</td>
<td>$d$</td>
</tr>
<tr>
<td>Radius of wheel [m]</td>
<td>$r$</td>
</tr>
<tr>
<td>Rotational speed of right wheel [rad/s]</td>
<td>$\phi_R$</td>
</tr>
<tr>
<td>Rotational speed of left wheel [rad/s]</td>
<td>$\phi_L$</td>
</tr>
<tr>
<td>Translational speed of right wheel [m/s]</td>
<td>$v_R$</td>
</tr>
<tr>
<td>Translational speed of left wheel [m/s]</td>
<td>$v_L$</td>
</tr>
<tr>
<td>Turning radius [m]</td>
<td>$\rho$</td>
</tr>
<tr>
<td>Velocity [m/s]</td>
<td>$v$</td>
</tr>
<tr>
<td>Yaw rate [rad/s]</td>
<td>$w$</td>
</tr>
</tbody>
</table>

![Fig. 6 Skid-Steer Model](image)

3.2. **Simulation results**

The parameters of the simulation are as shown in Table III. Slippery ground and slope were used for the simulation environment as shown in Fig. 7. There are 30 moving obstacles in the simulation field shown as blue discs in the following figures in Fig. 8. We assume velocity vector of the moving obstacles to be already
known. Fig. 9 shows the generated paths as the result of planning and the vehicle trajectories as the result of control to trace the planned path. The vehicle passes through the slippery ground and the slope. Finally, the vehicle arrives at the goal area while repeating the replan by SAP. The vehicle hardly steers so as not to spin out in slippery ground. And Drive power is always above grade resistance in diagonally running for the slope.

3.3. Experimental result

The parameters of the experiment are as shown in Table. V. Asphalt road is used for the experimental environment as shown in Fig. 10. There are five walking persons in the experimental field shown as blue and red discs in the following figures in Fig. 11. We assume velocity vector of the walking persons to be already known. The velocities of the obstacles are shown in Table. IV. $V_X$ means X-axial velocity, and $V_Y$ means Y-axial velocity. Fig. 12 shows the error of time estimated by SAP. The error is reset at the time planned again, because SAP gets actual time. Fig. 13 shows the generated paths as the result of planning and the vehicle trajectories as the result of control to trace the planned path. The vehicle arrives at the goal area while avoiding the five walking persons, considering kinematics and dynamics of the robot, and repeating the replanning by SAP.

<table>
<thead>
<tr>
<th>Walking persons</th>
<th>X (default position)</th>
<th>Y (default position)</th>
<th>$V_X$</th>
<th>$V_Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (blue disc)</td>
<td>50</td>
<td>3</td>
<td>-0.3</td>
<td>0</td>
</tr>
<tr>
<td>2 (blue disc)</td>
<td>40</td>
<td>1</td>
<td>-0.3</td>
<td>0</td>
</tr>
<tr>
<td>3 (blue disc)</td>
<td>30</td>
<td>5</td>
<td>-0.3</td>
<td>0</td>
</tr>
<tr>
<td>4 (red disc)</td>
<td>10</td>
<td>5</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>5 (red disc)</td>
<td>20</td>
<td>1</td>
<td>0.2</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Notation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>mass[kg]</td>
<td>m</td>
<td>1130</td>
</tr>
<tr>
<td>Front tire cornering power[N/rad]</td>
<td>$K_f$</td>
<td>30000</td>
</tr>
<tr>
<td>Rear tire cornering power[N/rad]</td>
<td>$K_r$</td>
<td>50000</td>
</tr>
<tr>
<td>Wheel base[m]</td>
<td>l</td>
<td>2.6</td>
</tr>
<tr>
<td>Distance from center of gravity to front axle[m]</td>
<td>$l_f$</td>
<td>1</td>
</tr>
<tr>
<td>Distance from center of gravity to rear axle[m]</td>
<td>$l_r$</td>
<td>1.6</td>
</tr>
<tr>
<td>Drive power[N]</td>
<td>$F_{pwr}$</td>
<td>3913</td>
</tr>
<tr>
<td>Air resistance coefficient</td>
<td>$C_d$</td>
<td>0.01</td>
</tr>
<tr>
<td>Vehicle height[m]</td>
<td>H</td>
<td>0.5</td>
</tr>
<tr>
<td>Tread[m]</td>
<td>T</td>
<td>1.48</td>
</tr>
<tr>
<td>Acceleration of gravity[m/s²]</td>
<td>g</td>
<td>9.8</td>
</tr>
<tr>
<td>Rolling resistance coefficient</td>
<td>$C_w$</td>
<td>0.015</td>
</tr>
<tr>
<td>Air density[kg/m³]</td>
<td>$\rho$</td>
<td>1.29</td>
</tr>
</tbody>
</table>
4. Conclusion

In this paper, we presented simultaneous adaptive path planning (SAP). We described that path planner of the rover should consider various problems, i.e., evaluation whether robot can accomplish the states such as position and posture, consideration of kinematics and dynamics of the robot, avoidance of the moving obstacles, and estimate of arrival time. We proved that SAP is able to adapt to these problems by simulation and experiment. It is expected that the rover can explore while adapting to kinematic and dynamic constraints, and dynamically changing environment simultaneously by SAP.

5. References