2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Switching Sampling Space of Model Predictive Path-Integral Controller to Balance Efficiency and Safety in 4WIDS Vehicle Navigation

Mizuho Aoki[1], Kohei Honda[1], Hiroyuki Okuda[1], and Tatsuya Suzuki[1]

***Abstract*— Four-wheel independent drive and steering vehicle (4WIDS Vehicle, Swerve Drive Robot) has the ability to move in any direction by its eight degrees of freedom (DoF) control inputs. Although the high maneuverability enables efficient navigation in narrow spaces, obtaining the optimal command is challenging due to the high dimension of the solution space. This paper presents a navigation architecture using the Model Predictive Path Integral (MPPI) control algorithm to avoid collisions with obstacles of any shape and reach a goal point. The key idea to make the problem easier is to explore the optimal control input in a reasonably reduced dimension that is adequate for navigation. Through evaluation in simulation, we found that the selecting sampling space of MPPI greatly affects navigation performance. In addition, our proposed controller which switches multiple sampling spaces according to the real-time situation can achieve balanced behavior between efficiency and safety. Source code is available at `https://github.com/MizuhoAOKI/mppi_swerve_drive_ros`.**

I. INTRODUCTION

Four-wheel independent drive and steering (4WIDS) vehicles have the potential to enhance the level of vehicle motion as a next-generation power transmission system for automobiles [1] [2]. 4WIDS is capable of independently controlling the steering angle and drive torque of all four wheels, enabling holonomic movements such as pivoting in place and diagonal movement. In particular, unlike other types of holonomic robots (*e.g.*, omnidirectional wheels and mecanum wheels robots), steerable vehicles can drive on rough terrain and maintain high-speed stability [2]–[4]. However, due to its high-dimensional input space (*i.e.*, eight degrees of freedom), control can be challenging. In addition, smooth steering control is required to avoid mechanical failures.

In general, gradient-based Model Predictive Control (MPC) is one of the effective approaches for redundant systems, where the input dimension is larger than the state dimension [3], [5], [6]. However, gradient-based MPC has challenges such as the inability to handle non-differentiable cost functions (*e.g.*, cost map) or the possibility of converging to local minima in nonlinear problems. Therefore, sample-based MPC approaches that do not rely on gradients, such as the Model Predictive Path Integral control (MPPI) [7], are practically promising and used for a wide variety of applications [8]–[17].

When applying MPPI to redundant systems, the dimension of the input space is a critical issue. The larger the dimension of the input space, the more pronounced the curse of dimensionality becomes. This means that the number of samples required to adequately cover the input space increases exponentially, resulting in a significant decrease in sampling efficiency. How to avoid the curse of dimensionality and apply MPPI efficiently remains a critical issue in controlling redundant systems.

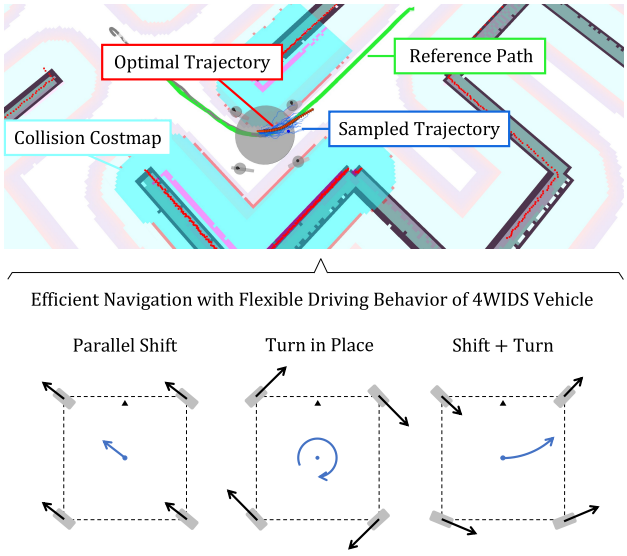


Fig. 1: 4WIDS vehicles can achieve various types of motion, such as moving diagonally and turning in place. Our navigation architecture makes good use of these capabilities to achieve efficient and stable navigation in narrow spaces avoiding obstacles.

In this paper, we focus on analytically constraining the input space of 4WIDS through geometric constraints, allowing for the application of MPPI to a reduced-dimensional input space. As a result, we can avoid the aforementioned curse of dimensionality and maintain the sample efficiency of MPPI while controlling 4WIDS. To the best of our knowledge, this is the first study to apply MPPI to 4WIDS.

The primary contribution of this work is three-fold. First, we derive that it is possible to reduce the original eight-dimensional input space of 4WIDS to three dimensions by applying static geometrical constraints. However, our experiments revealed an insight that the three-dimensional control input space degrades the stability of navigation.

Therefore, as a second contribution, we demonstrate that *controlling 4WIDS is more effective with a slightly redundant input space than with the bare minimum of three dimensions*. Specifically, we opted to apply MPPI to a four-dimensional input space that includes redundancy, instead of sticking to

*This work was not supported by any organization.

[1]The Department of Mechanical Systems Engineering, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi, Japan, mizuhoaoki1998@gmail.com

the minimal three dimensions. As a result, our empirical simulation results proved to enhance the success rate of navigation tasks.

Third, as a final contribution, we propose an approach to switch the control input space in real-time, depending on the situation. The hybrid sampling showed balanced performance to achieve both efficiency and a high success rate in navigation tasks.

II. RELATED WORK

A. Conventional Control Approaches for 4WIDS

A simple way to control a 4WIDS vehicle is by limiting its degrees of freedom with adding constraints. In [18], a constraint is added that the front steering angle is the negative of the rear steering angle and applied pure-pursuit algorithm for path tracking. This approach is easy to understand and computationally efficient, but it is not suitable for utilizing the vehicle's maneuverability.

Fuzzy logic is a powerful tool to handle full vehicle input space by preparing multiple driving modes and switching them according to the situation [19]. However, it requires a high level of domain knowledge to design each mode and the predefined rules are difficult to deal with unknown environments.

To guarantee the stability of the system in any situation, the Lyapunov function is used to design the control law [20], [21]. Stable path tracking can be achieved, but considering other tasks such as smooth steering and obstacle avoidance is challenging for the framework.

Since considering various goals and constraints can broaden the scope of applications, Model Predictive Control (MPC) has been widely used, which predicts the future state of the target system and gets the optimal control input minimizing a cost function [3], [5], [6]. Despite its high flexibility, MPC is computationally expensive. To achieve real-time control, using a gradiant-based optimization solver, linear approximation of the vehicle model, or assuming an accurate reference trajectory are practical strategies.

However, to perform practical local planning that smoothly avoids obstacles of arbitrary shape, complex optimization problems including non-convex and non-differentiable formulation are essential, which is difficult to handle with conventional methods. This paper introduces a sampling-based solver to expand the scope of MPC-based approaches and aims to achieve more general navigation tasks in cluttered environments.

B. Applications of Sampling-based MPC for Redundant Systems

Sampling-based MPC is a powerful tool to control redundant systems and has been applied to various complex control problems.

The simplest algorithm is to sample several control sequences and evaluate them to find the best one. Complex systems such as humanoid robots and quadrupedal robots can be controlled by a simple algorithm [17]. However, this approach has low sample efficiency and the behavior is likely to be jerky.

Using the MPPI algorithm can enhance the sample efficiency and generate smoother behavior. Manipulators with high degrees of freedom can successfully avoid obstacles [16], and stable manipulation is achieved even when the system model is uncertain [15]. Combination with reinforcement learning has also been studied [22]. Learning environment dynamics and exploring the solution in the latent space successfully operated a 38-DoF control target.

While the application of MPPI is expanding, there is still little knowledge about how to choose the sampling space to explore the solution effectively. This paper aims to provide a new insight into the sampling space selection with comparative experiments and theoretical analysis.

III. TARGET TASK DESCRIPTION

The goal of our system is to navigate a 4WIDS vehicle to a given goal while avoiding collision with surrounding obstacles (Fig. 2). We assume that a global planner generates a reference trajectory, which is a sequence of positions and orientations in a 2D plane. Therefore, the main focus of our study is to design a local planner that generates a vehicle motion to follow the reference trajectory while safely avoiding obstacles. The local planner sends an eight DoF vehicle command consisting of a wheel speed and a steering angle for each wheel to the vehicle actuators. Especially in our navigation environment where the obstacles are densely distributed, the high maneuverability of 4WIDS vehicles, such as small turning radius and diagonal movement, is key to achieving smooth and efficient driving.

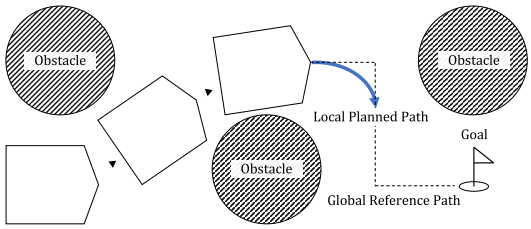


Fig. 2: Navigation task

IV. MODELING OF 4WIDS VEHICLE MOTION

To achieve accurate vehicle control, it is important to understand the characteristics of vehicle motion. In this section, we formulate the model of 4WIDS vehicle behavior, mainly used for sampling future vehicle poses in our model predictive controller.

A. 4WIDS Vehicle Dynamics

Although it is a straightforward approach to model vehicle behavior using Newton's laws of motion, our study does not focus on this approach. To formulate the full vehicle dynamics, it is necessary to consider the eight tire forces generated by the four wheels, as shown in Fig.3. However, to obtain the forces, it is necessary to observe tire slip angles, tire slip ratios, and tire physical parameters, which are not easy to

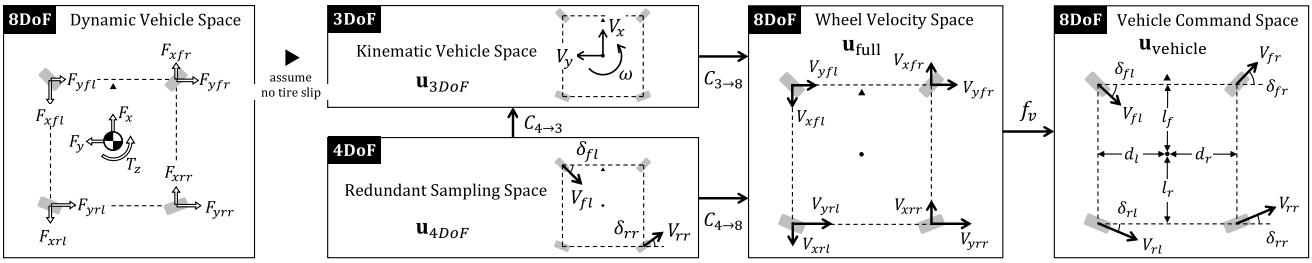


Fig. 3: Relationship between spaces; Dynamics(8DoF) can be simplified to Kinematics(3DoF) with assuming no tire slip. Two types of sampling space are tested in this study; $\mathbf{u}_{3\text{DoF}}$ and $\mathbf{u}_{4\text{DoF}}$ to compare the navigation performance. Both spaces can be converted to the vehicle command space $\mathbf{u}_{\text{vehicle}}$ with conversion matrix $C_{n\to 8}(n \in \{3,4\})$ and nonlinear projection f_v.

measure accurately in real-world applications [23]. Even if we have true dynamics and explore the full eight-dimensional input space, most of the solutions will include large tire slips, which cause unstable vehicle behavior. Therefore, predicting vehicle motion with full dynamics is not always practical, especially for real-world applications.

B. 4WIDS Vehicle Kinematics

An efficient way to easily model the vehicle motion is to focus on vehicle kinematics [21]. Essentially, the difficulty of controlling 4WIDS vehicles comes from the redundancy of the control degrees of freedom, which means that there are multiple solutions to achieve a certain vehicle behavior. The kinematic formulation adds assumptions that the vehicle is moving at a constant velocity, and the tire slip is negligible. The assumptions can be interpreted as a reduction of the control input space and focusing on considering more practical vehicle motion.

Under the kinematic assumptions, the vehicle's motion can be simplified as a model with three degrees of freedom $\mathbf{u}_{3\text{DoF}}$; longitudinal velocity V_x, lateral velocity V_y, and angular velocity ω of the vehicle center. The relationship between the vehicle center velocity $\mathbf{u}_{3\text{DoF}}$ and the eight-wheel velocities $\mathbf{u}_{\text{full}}$ is formulated as follows and shown in Fig.3.

$$\mathbf{u}_{3\text{DoF}} = [V_x, V_y, \omega]^T, \tag{1}$$

$$\mathbf{u}_{\text{full}} = [V_{xfl}, V_{xfr}, V_{xrl}, V_{xrr}, V_{yfl}, V_{yfr}, V_{yrl}, V_{yrr}]^T, \tag{2}$$

$$\mathbf{u}_{\text{full}} = \mathbf{C}_{3\to 8}\ \mathbf{u}_{3\text{DoF}}, \tag{3}$$

$$\mathbf{C}_{3\to 8} = \begin{bmatrix} 1 & 0 & -d_l \\ 1 & 0 & d_r \\ 1 & 0 & -d_l \\ 1 & 0 & d_r \\ 0 & 1 & l_f \\ 0 & 1 & l_f \\ 0 & 1 & -l_r \\ 0 & 1 & -l_r \end{bmatrix}. \tag{4}$$

The wheel velocity $\mathbf{u}_{\text{full}}$ can be converted to the vehicle command space $\mathbf{u}_{\text{vehicle}}$ consisting of four-wheel steering angles δ_* and four-wheel velocities V_* for the front left (*fl*), front right (*fr*), rear left (*rl*), and rear right (*rr*) wheels. A nonlinear transformation function f_v is used for the conversion,

$$\mathbf{u}_{\text{vehicle}} = [\delta_{fl}, \delta_{fr}, \delta_{rl}, \delta_{rr}, V_{fl}, V_{fr}, V_{rl}, V_{rr}]^T, \tag{5}$$

$$\mathbf{u}_{\text{vehicle}} = f_v(\mathbf{u}_{\text{full}}), \tag{6}$$

$$\delta_* = \arctan\frac{V_{y*}}{V_{x*}}, \tag{7}$$

$$V_* = \sqrt{V_{x*}^2 + V_{y*}^2}, \tag{8}$$

where $*$ represents the wheel positions *fl*, *fr*, *rl*, *rr*.

Note that the kinematic formulation only requires the geometric relationship of the tires which is easy to obtain, and is easily applicable to real-world applications.

C. Exploring Solution in Redundant Control Input Space

Sampling based controller needs to get a variety of solutions from the control input space $\mathbf{u}_{3\text{DoF}}$. In this study, we found that exploring a slightly redundant space $\mathbf{u}_{4\text{DoF}}$ and converting it to $\mathbf{u}_{3\text{DoF}}$ was more effective in achieving smooth and stable vehicle motion with MPPI. This idea is inspired by methods such as Koopman Operator [24] and Dynamic Mode Decomposition [25], which expand the state space to express the complex dynamics as a linear system in a higher dimensional space.

Since $\mathbf{u}_{4\text{DoF}}$ is a 4-dimensional redundant space, it does not always satisfy kinematic constraints formulated in Eq.(3). Therefore, projection matrix $\mathbf{C}_{4\to 8}$ is used to map $\mathbf{u}'_{4\text{DoF}}$ to $\mathbf{u}_{\text{vehicle}}$, so that the sampled vehicle motion always satisfies the kinematic constraints.

$$\mathbf{u}_{4\text{DoF}} = [V_{fl}, V_{rr}, \delta_{fl}, \delta_{rr}]^T, \tag{9}$$

$$\begin{aligned}\mathbf{u}'_{4\text{DoF}} &= f(\mathbf{u}_{4\text{DoF}}) \\ &= [V_{fl}\cos\delta_{fl}, V_{rr}\cos\delta_{rr}, V_{fl}\sin\delta_{fl}, V_{rr}\sin\delta_{rr}]^T \\ &= [V_{xfl}, V_{xrr}, V_{yfl}, V_{yrr}]^T,\end{aligned} \tag{10}$$

$$\mathbf{u}_{\text{vehicle}} = \mathbf{C}_{4\to 8}\mathbf{u}'_{4\text{DoF}}, \tag{11}$$

$$\mathbf{C}_{4\to 8} = \mathbf{C}_{3\to 8}\mathbf{C}_{4\to 3}, \tag{12}$$

$$\mathbf{C}_{4\to 3} = \begin{bmatrix} \frac{d_r}{d_l+d_r} & \frac{d_l}{d_l+d_r} & 0 & 0 \\ 0 & 0 & \frac{l_r}{l_f+l_r} & \frac{l_f}{l_f+l_r} \\ \frac{-1}{2(d_l+d_r)} & \frac{1}{2(d_l+d_r)} & 0 & 0 \end{bmatrix}. \tag{13}$$

The conversion to reduce a dimention in Eq.(13) is based on the idea of taking the average of the control inputs between front wheel space (V_{fl}, δ_{fl}) and the rear wheel space (V_{rr}, δ_{rr}) to obtain the compromised vehicle center angular velocity ω.

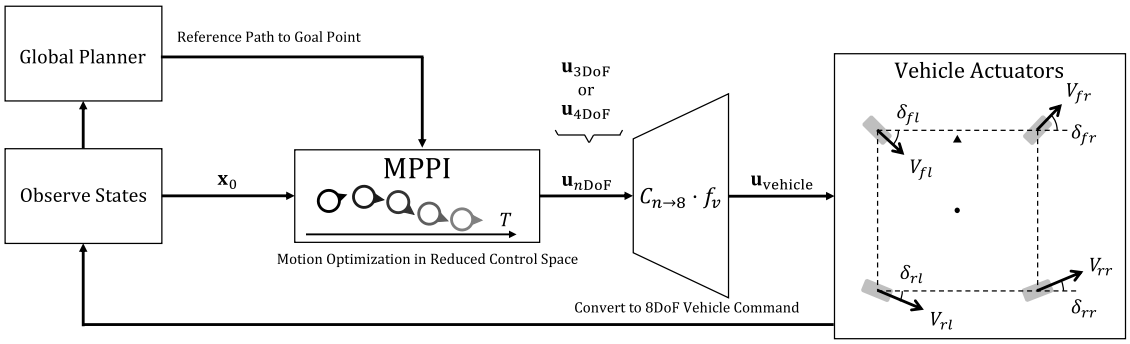


Fig. 4: Overview of the control architecture (See Section V-A). (a) global planner generates a reference trajectory based on current vehicle pose and the given map, (b) MPPI generates the optimal control input in a reduced dimensional space, (c) the n DoF ($n \in \{3,4\}$) control input is converted to the 8DoF vehicle command space, and (d) the vehicle actuators execute the command.

V. NAVIGATION ARCHITECTURE FOR 4WIDS VEHICLE

A. System Overview

The navigation system for the 4WIDS vehicle is composed of four main components below in this work (See Fig.4)

1) State Observation: The map of the environment is given, and the vehicle localizes itself using 2D LiDAR point cloud data and odometry information.

2) Global Path Planning: The global planner calculates a path from the current vehicle position to the goal using Dijkstra's algorithm.

3) Local Path Planning: In this work, a Model Predictive Path-Integral Controller (MPPI) is used to plan the local path and calculate the next optimal control input. Since exploring the full vehicle command space which has eight degrees of freedom is difficult and inefficient, MPPI samples solutions in a dimension that is reasonably reduced by kinematic constraints such as $\mathbf{u}_{3\text{DoF}}$ and $\mathbf{u}_{4\text{DoF}}$ defined in Section IV.

4) Vehicle Command Calculation: Since MPPI calculates the control input in a reduced dimension, the control input is converted to the 8DoF vehicle command and sent to the vehicle actuators. The optimal control input calculated by MPPI is converted to the 8DoF vehicle command and sent to the actuators. The conversion is done by Eq.(4), (13) and (8) in Section IV.

B. Algorithm of MPPI Controller

In this study, local planning is performed using the Model Predictive Path-Integral (MPPI) Controller shown in Algorithm 1. MPPI is an optimal control algorithm that uses a sample-based approach to compute the optimal control input sequence in the near future. The algorithm is based on the idea of sampling the control input sequences as normal distributions centered at the previous optimal input sequence, and getting the optimal sequence as weighted sum so that better sequences are heavily weighted and vice versa.

For a discrete-time, continuous state-action system $\mathbf{x}_t = \mathbf{F}(\mathbf{x},\mathbf{u})$, K samples of input sequences $V = \{\mathbf{v}_t\}_{t=0}^{T-1}$ are generated by adding Gaussian noise to mean control input sequence $U = \{\mathbf{u}_t\}_{t=0}^{T-1}$ with covariance matrix Σ. After preparing the samples, the optimal control input sequence is easily obtained by calculating the weighted sum of the samples. The stage cost function $c(\mathbf{x},\mathbf{u})$ and terminal cost function $\phi(\mathbf{x})$ are defined to evaluate the quality of the samples. Let S_k be the total cost of the k-th sample whose input sequence is V_k, the weight for the sequence is

$$w(V_k) = \frac{1}{\eta}\left(-\frac{1}{\lambda}S(V_k) + \lambda \sum_{\tau=0}^{T-1} \mathbf{u}_t^T \Sigma^{-1} \mathbf{v}_t - \rho\right) \tag{14}$$

where η is the normalization factor, λ is the constant temperature parameter, and ρ is the minimum cost among the samples.

The notable benefits of MPPI are that it can be applied to a wide range of optimization problems, including non-linear, non-convex, and non-differentiable cost functions and system models. The algorithm is theoretically guaranteed to minimize the forward Kullback-Leibler (KL) divergence between the proposed distribution and the optimal distribution which minimizes the total cost function.

C. MPPI Switching Multiple Control Spaces

Through analysis of the experimental results in Section VI-D, we found that selecting the control input space to explore the solution affects the navigation performance significantly. To take advantage of the strengths and mitigate the weaknesses of each control input space, we propose a method to switch between multiple spaces in real time according to the situation explained in Algorithm 2. If two control input spaces are defined as $\mathbf{u}^A$ and $\mathbf{u}^B$, both spaces need to update the control input sequence in each time step as a preparation for the next calculation. The conversion functions ConvertToSpaceA and ConvertToSpaceB are needed to convert the control input sequence to the other space. If we switch $\mathbf{u}_{3\text{DoF}}$ to $\mathbf{u}_{4\text{DoF}}$, the conversions are done using Eq.(4), (13), and (8) in Section IV.

Algorithm 1 Model Predictive Path-Integral Controller

Given: **F**, g: Transition Model;
K: Number of samples;
T: Number of timesteps;
$U \leftarrow (\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{T-1})$: Initial control sequence;
$\Sigma, \phi, c, \gamma, \lambda, \alpha, \Delta t$: Cost functions and parameters;
while task not completed **do**
  $\mathbf{x}_0 \leftarrow$ ObserveSystemState()
  for k = 0 to K-1 **do**
    $\mathbf{x} \leftarrow \mathbf{x}_0$;
    Sample $\mathcal{E} = (\varepsilon_0^{\mathrm{k}} \ldots \varepsilon_{\mathrm{T}-1}^{\mathrm{k}}),\ \varepsilon \in \mathcal{N}(0, \Sigma)$;
    for t = 1 to T-1 **do**
      if $k < (1-\alpha)K$ **then**
        $\mathbf{v}_{t-1} = \mathbf{u}_{t-1} + \varepsilon_{t-1}^k$; ▷ samples for exploitation
      else
        $\mathbf{v}_{t-1} = \varepsilon_{t-1}^k$; ▷ samples for exploration
      $x \leftarrow \mathbf{F}(\mathbf{x}, g(\mathbf{v}_{t-1}), \Delta t)$;
      $S_k += c(\mathbf{x}, \mathbf{u}) + \gamma \mathbf{u}_{t-1}^T \Sigma^{-1} \mathbf{v}_t$ ▷ add stage cost
    $S_k += \phi(\mathbf{x})$ ▷ add terminal cost
  $\rho \leftarrow \min_k[S_k]$;
  $\eta \leftarrow \sum_{k=1}^K \exp\left(-\frac{1}{\lambda}(S_k - \rho)\right)$;
  for k = 0 to K-1 **do**
    $\mathbf{w}_k \leftarrow \frac{1}{\eta} \exp\left(-\frac{1}{\lambda} S_k\right)$; ▷ calculate sample weights
  for t = 0 to T-1 **do**
    $U \leftarrow U + \left(\sum_{k=1}^K \mathbf{w}_k \mathcal{E}^k\right)$; ▷ calculate weighted sum
  $\mathbf{u}_{\text{vehicle}} \leftarrow$ ConvertTo8DoFVehicleCommand($\mathbf{u}_0$);
  SendToVehicleActuators($\mathbf{u}_{\text{vehicle}}$);
  for t = 1 to T-1 **do**
    $\mathbf{u}_{t-1} \leftarrow \mathbf{u}_t$; ▷ shift the control sequence
  $\mathbf{u}_{T-1} \leftarrow$ Initialize($\mathbf{u}_{\mathrm{T}-1}$);

Algorithm 2 MPPI Switching Multiple Control Input Spaces

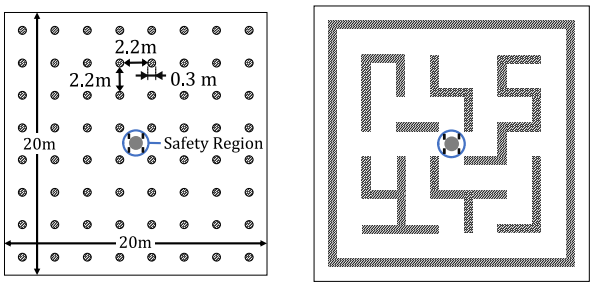
$U_0^A \leftarrow (u_0^A, \ldots, u_{T-1}^A)$: Initial control sequence of space A;
$U_0^B \leftarrow (u_0^B, \ldots, u_{T-1}^B)$: Initial control sequence of space B;
while task not completed **do**
  mode $\leftarrow$ SelectMode(); ▷ select control input space
  if mode is A **then**
    $U_{t+1}^A \leftarrow$ SolveMPPI($\mathrm{U_t^A}$);
    $U_{t+1}^B \leftarrow$ ConvertToSpaceB($\mathrm{U_{t+1}^A}$);
  else if mode is B **then**
    $U_{t+1}^B \leftarrow$ SolveMPPI($\mathrm{U_t^B}$);
    $U_{t+1}^A \leftarrow$ ConvertToSpaceA($\mathrm{U_{t+1}^B}$);

VI. EXPERIMENTS

A. Simulation Setup

To evaluate the navigation performance of the proposed architecture, we set up a simulation environment using Gazebo simulator. Two types of fields are prepared, "Cylinder Garden" and "Maze" (Fig. 5). The vehicle need to reach 10 goals sequentially in a episode as fast as possible while avoiding densely placed obstacles. The four wheel positions are set symmetrically as $l_f = l_r = d_l = d_r = 0.5$ [m] (Fig. 3).

Our evaluation system was developed using ROS and C++. Calculation is performed on a desktop computer with an Intel Core i7-13700KF CPU and 32GB of RAM. For faster computation, we used CPU multi-threading with OpenMP



(a) Cylinder Garden (Easier) **(b)** Maze (Harder)

Fig. 5: Simulation environment

[26].

B. Cost Formulation for MPPI

In this section, the cost functions $c(\mathbf{x}, \mathbf{u})$ and $\phi(\mathbf{x})$ (See Algorithm 1 are defined for the MPPI controller.

The stage cost $c(\mathbf{x}, \mathbf{u})$ is defined as the weighted sum of the following terms,

$$c(\mathbf{x}, \mathbf{u}) = 40\ c_{\text{dist}}(p_x, p_y) + 30\ c_{\text{angle}}(\theta) + 10\ c_{\text{speed}}(v) + 50\ c_{\text{collision}}(p_x, p_y) + c_{\text{cmd}}(\mathbf{u}), \tag{15}$$

where p_x and p_y are the vehicle's position, θ is the vehicle's yaw angle, and v is the vehicle's velocity. $c_{\text{dist}}(p_x, p_y)$, $c_{\text{angle}}(\theta)$, and $c_{\text{speed}}(v)$ are quadratic error from the reference path and constant target velocity v_{des}, respectively. $c_{\text{collision}}(p_x, p_y)$ is a binary cost function that returns 1 if the vehicle is in collision, and 0 otherwise;

$$c_{\text{collision}}(p_x, p_y) = \begin{cases} 0 & \text{if no collision} \\ 1 & \text{if collision} \end{cases}. \tag{16}$$

$c_{\text{cmd}}(\mathbf{u})$ is used to smooth the vehicle actuator commands. In either control space $\mathbf{u}_{\text{3DoF}}$ or $\mathbf{u}_{\text{4DoF}}$ to be explored, it can be converted to 8-dimensional vehicle actuator commands $\mathbf{u}_{\text{vehicle}}$ with equations Eq.(3) and (11). With previous vehicle command $\mathbf{u}_{\text{vehicle}}^{\text{prev}}$ in the control sequence, minimizing the penalty term

$$c_{\text{cmd}}(\mathbf{u}) = \|\mathbf{u}_{\text{vehicle}} - \mathbf{u}_{\text{vehicle}}^{\text{prev}}\|_2 \tag{17}$$

can smooth the vehicle actuator commands.

the terminal cost $\phi(\mathbf{x})$ is added only for preventing reverse driving,

$$\phi(\mathbf{x}) = 50\ \phi_{\text{goal}}(p_x, p_y), \tag{18}$$

where $\phi_{\text{goal}}(p_x, p_y)$ is the quadratic error from the goal position.

C. Preparing MPPI Controllers for Comparison

To investigate the characteristics of the MPPI controller depending on the choice of the control space, MPPI-3D and MPPI-4D are prepared to explore two different control spaces, $\mathbf{u}_{\text{3DoF}}$ and $\mathbf{u}_{\text{4DoF}}$.

Since the variance parameter affects the controller's behavior, setting the variance of the control space carefully is important for fair comparison. Changing the variance

TABLE I: Evaluation Results of 100 Navigation Episodes
blue value is the best score, and red value is the worst score of all four controllers.

Field	Cylinder Garden				Maze			
Method Control Space	MPPI-3D(a) $[V_x, V_y, \omega]$	MPPI-3D(b) $[V_x, V_y, \omega]$	MPPI-4D $[V_{fl}, V_{rr}, \delta_{fl}, \delta_{rr}]$	MPPI-H 3D(a) / 4D	MPPI-3D(a) $[V_x, V_y, \omega]$	MPPI-3D(b) $[V_x, V_y, \omega]$	MPPI-4D $[V_{fl}, V_{rr}, \delta_{fl}, \delta_{rr}]$	MPPI-H 3D(b) / 4D
Cost [-] ↓	3241.7	1900.5	1455.8	2425.4	10030.4	3918.8	2452.3	2887.6
Calc. Time [ms] ↓	24.1	23.0	27.6	26.6	19.7	19.9	24.0	21.0
Steering Rate [rad/s] ↓	4.5	3.1	3.6	4.0	6.0	3.6	5.0	3.5
Wheel Acc. [m/s^2] ↓	5.03	3.36	4.08	4.98	6.02	3.77	4.85	4.02
Traj. Length [m] ↓	51.9	46.0	40.8	42.6	72.1	64.8	55.2	55.3
Episode Time [s] ↓	36.4	41.3	38.4	31.2	49.6	55.9	52.1	44.8
Success Rate [%] ↑	76	89	100	99	33	58	98	96

TABLE II: MPPI Params

Param	Value	Unit
K	3000	sample
T	30	step
Δt	0.033	sec
α	0.1	-
λ	250	-
γ	6.25	-

TABLE III: Controller Params

Param	Value	Unit
Control Interval Δt_i	0.05	sec
Target Velocity v_{des}	2.00	m/s
Max. Velocity $v_{\max}$	2.00	m/s
Max. Yawrate $\omega_{\max}$	1.58	rad/s
Max. Steering Angle	1.58	rad

TABLE IV: MPPI Variance Params

Name	Control Space	Variance Σ
MPPI-3D(a)	$[V_x, V_y, \omega]$	$[1.00, 1.00, 0.78]$
MPPI-3D(b)	$[V_x, V_y, \omega]$	$[0.55, 0.55, 0.96]$
MPPI-4D	$[V_{fl}, V_{rr}, \delta_{fl}, \delta_{rr}]$	$[1.00, 1.00, 0.78, 0.78]$

parameters, two types of MPPI-3D are prepared, MPPI-3D(a) and MPPI-3D(b). The variance parameters of MPPI-3D(a) and MPPI-4D are set systematically as shown in Table IV following the rule that the variance is half of the maximum value of the control space defined in Table III. This consideration is to explore a wide range of the control space, as the normal distribution contains about 95% of the data within twice the standard deviation.

Another approach is to fit a normal distribution numerically close to the sampled results of MPPI-4D. MPPI-3D(b) follows this procedure, and the variance parameters are calculated with maximum likelihood estimation of the normal distribution, when the vehicle is stopped and all the steering angles are zero in the space of $\mathbf{u}_{\text{4DoF}}$.

Additionally, a hybrid MPPI controller, MPPI-H, is prepared to switch between MPPI-3D and MPPI-4D depending on the situation. From the verification results, we determined that the mode selection should be based on the target path tracking error. The mode switching function in Algorithm 2 is defined as follows with constant parameter $d_{\text{thresh}} = 0.3$ [m] and $\theta_{\text{thresh}} = 0.3$ [rad],

$$\text{SelectMode}() = \begin{cases} \mathbf{u}_{\text{3DoF}} & \text{if } c_{\text{dist}}(p_x, p_y) < d_{\text{thresh}} \text{ and } c_{\text{angle}}(\theta) < \theta_{\text{thresh}} \\ \mathbf{u}_{\text{4DoF}} & \text{otherwise} \end{cases}. \tag{19}$$

D. Definition of Evaluatioin Metrics

Here the evaluation metrics are defined to compare the performance of the MPPI controllers. All metrics are calculated for all episodes, and the mean value is used for evaluation.

1) Cost: "Cost" is the sum of stage cost and terminal cost of optimal trajectory output from MPPI. This metric indicates how well the MPPI controller can minimize the cost function and get close to the optimal behavior.

2) Vehicle Command Change: To evaluate the smoothness of the vehicle actuator commands, two metrics are defined. "Steering Rate" is the absolute value of the steering angular velocity. "Wheel Acceleration" means the absolute change of the wheel velocity. For both metrics, the mean values of four wheels are used for evaluation.

3) Navigation Efficiency: two metrics are defined to evaluate the navigation efficiency. "Trajectory Length" is selected to evaluate how the vehicle could reach the goal with a short path. "Episode Time" is also used to know how fast the vehicle could reach the goal.

4) Success Rate: Success rate is the percentage of episodes that the vehicle reached all the given goal points. Collision with obstacles and getting stuck in the field are major factors of failure.

E. Evaluation Results Comparison

100 episodes of navigation are performed for each field, and the evaluation results are shown in Table I. For each MPPI controller, the mean calculation time is less than 30ms and works in real-time with the control interval $\Delta t_i = 50$ [ms]. Even though the cost function and algorithmic parameters are common, the results drastically changed by exploring different control spaces.

MPPI-3D(a) has a short episode time, which means it can drive efficiently. A lower success rate (76% in Cylinder Garden, 33% in Maze) means that it fails to complete episodes frequently, and has difficulty in safe and stable navigation. Since MPPI-3D(b) has a smaller variance in the vehicle velocity space than MPPI-3D(a), it moves slowly and appears more conservative behavior. The episode time is the worst among the four controllers in both fields. However, the success rate is only slightly improved (89% in Cylinder Garden, 58% in Maze) from MPPI-3D(a) and still has low stability.

On the other hand, MPPI-4D showed an extremely high success rate (100% in Cylinder Garden, 99% in Maze) and

stable navigation behavior. Cost and trajectory length are the best among the four controllers, it means that MPPI-4D is good at finding the optimal solution. Comparing the episode time in Cylinder Garden, MPPI-4D (38.4s) is faster than MPPI-3D(b) (41.3s), but MPPI-3D(a) (36.4s) surpasses MPPI-4D in terms of driving efficiency.

Summarizing the characteristics of MPPI-3D and MPPI-4D, MPPI-3D is good at high-speed driving but has low stability, and MPPI-4D is more conservative and is good at stable navigation but has lower efficiency. Then it is reasonable to switch control spaces depending on the situation to balance the efficiency and stability. In the case the vehicle is in a difficult situation (i.e. the target path tracking error is large), MPPI-4D is selected to ensure stable navigation. Otherwise, MPPI-3D is selected to drive efficiently.

As a result, the hybrid MPPI-H switching control spaces showed the best episode time (31.2s in Cylinder Garden, 44.8s in Maze), keeping the high success rate (99% in Cylinder Garden, 96% in Maze) compared to MPPI-4D. The fact that MPPI-H has no worst score in any of the metrics is also evidence that MPPI-H has a balanced performance.

F. Trajectory Comparison

To understand the characteristics of the controllers more deeply, the trajectories of the four controllers are compared in Fig. 6. This example scenario is a hard situation where the vehicle receives the next goal point and needs to turn sharply.

In Fig. 6, the MPPI-3D(b) turns with a larger radius than the other controllers, and a part of the trajectory is close to the collision with surrounding obstacles. It is a typical dangerous behavior and shows the reason of the lower success rate of MPPI-3D controllers.

On the other hand, MPPI-4D can turn in a small radius and run safely keeping a distance from the obstacles, showing the reason of the high success rate of MPPI-4D.

In the situation where the vehicle should turn sharply, MPPI-H activates MPPI-4D to drive carefully, resulting in the same turning behavior as MPPI-4D.

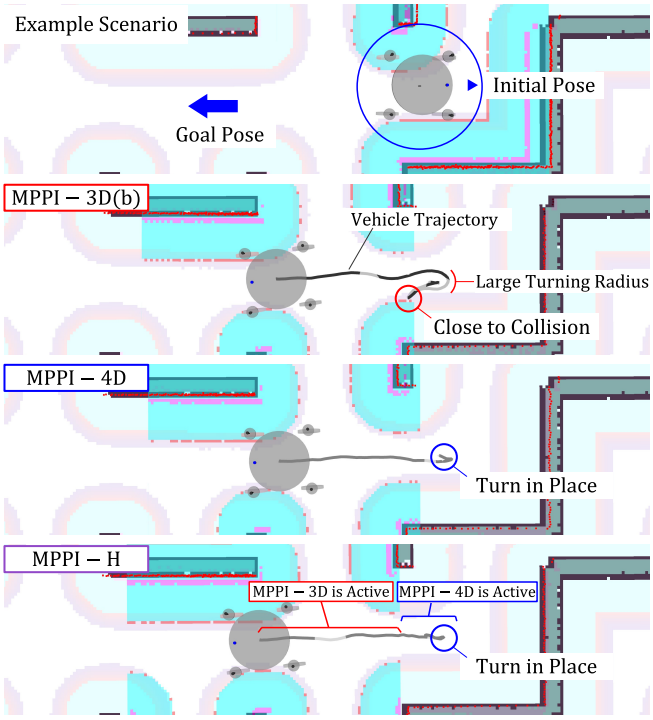


Fig. 6: Trajectory comparison of a difficult situation. While MPPI-3D shows dangerous behavior close to the collision, MPPI-4D and MPPI-H drive safely with turning in a small radius.

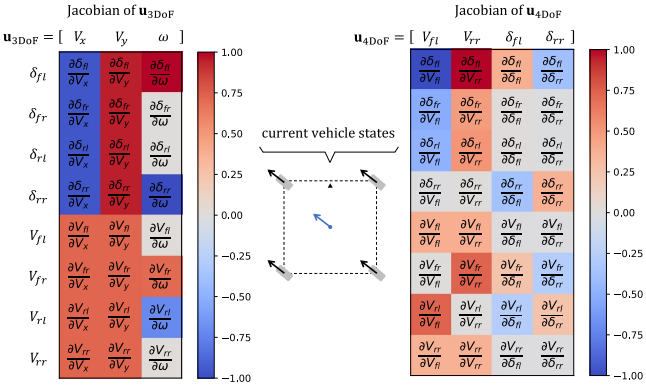


Fig. 7: Jacobian matrices comparison between $\mathbf{u}_{3DoF}$ and $\mathbf{u}_{4DoF}$ when $\delta_{fl} = \delta_{fr} = \delta_{rl} = \delta_{rr} = 0.25\pi$ [rad], $V_{fl} = V_{fr} = V_{rl} = V_{rr} = 0.7$ [m/s]. J_B has a more sparse structure than J_A, which makes the optimal solution search easier.

VII. DISCUSSION

In this section, we discuss why the performance changes by selecting control input space for sampling. Specifically, explanation of why the solution search in the control space $\mathbf{u}_{4DoF}$ is more likely to find a more optimal solution than in $\mathbf{u}_{3DoF}$ is provided.

In the navigation task, one of the most difficult situation is when the next goal point is specified in the opposite direction to the vehicle's heading. In this case, the optimal behavior is to quickly decelerate the vehicle and rotate the vehicle body to the target direction.

Here we consider the Jacobian matrices J_A, J_B with respect for the projection from each control input space $\mathbf{u}_{3DoF}, \mathbf{u}_{4DoF}$ to the vehicle command space $\mathbf{u}_{vehicle}$. The Jacobian matrices are normalized to the range of -1 to 1 and plotted as color maps in Fig.7.

As for the control input space $\mathbf{u}_{3DoF}$, deceleration of the vehicle speed V_x is strongly affects the vehicle steering angles. Therefore, even if a control sequence that includes a rapid deceleration is sampled, it is likely to be penalized due to the large change in the steering angle. As a result, the deceleration behavior is less likely to occur and the vehicle tries to rotate with the current speed, which sometimes causes collision with obstacles.

On the other hand, as for the control input space $\mathbf{u}_{4DoF}$, the Jacobian matrix J_B has a more sparse structure than J_A. This means that the change in the vehicle speed and the

steering angles are relatively independent. This feature makes the optimal solution search easier, and more likely to find the better solution including rapid deceleration and rotation of the vehicle at the same time.

From the above discussion, it can be considered that the sparsity of the Jacobian matrix is one of the guidelines for selecting the control input space for MPPI.

VIII. CONCLUSION

This paper presented a navigation framework for four-wheel independent driving and steering (4WIDS) Vehicle based on the Model Predictive Path-Integral (MPPI) control method. For efficiently solving the optimal control problem for high-dimensional systems, two types of reasonably reduced control spaces are introduced. Evaluation results show that the control space which has a slightly redundant dimension than the bare minimum can achieve more stable navigation. In addition, our novel approach that switches the control space in real-time can achieve both efficiency and stability at a high level.

Although this work offered a new perspective on how to choose an effective sampling space, devising a systematic method to find it is the next step. In addition, implementing the proposed method on a real platform is needed to verify its effectiveness in real-world environments.

REFERENCES

[1] P. Hang and X. Chen, "Towards autonomous driving: Review and perspectives on configuration and control of four-wheel independent drive/steering electric vehicles," *Actuators*, vol. 10, no. 8, 2021. [Online]. Available: https://www.mdpi.com/2076-0825/10/8/184

[2] K. Kanjanawanishkul, "Omnidirectional wheeled mobile robots: Wheel types and practical applications," *International Journal of Advanced Mechatronic Systems*, vol. 6, p. 289, 02 2015.

[3] X. Zhang, Y. Xie, L. Jiang, G. Li, J. Meng, and Y. Huang, "Trajectory tracking of a 4wis4wid robot using adaptive receding horizon control based on neurodynamics optimization," in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 2019, pp. 565–570.

[4] E. H. Binugroho, A. Setiawan, Y. Sadewa, P. H. Amrulloh, K. Paramasastra, and R. W. Sudibyo, "Position and orientation control of three wheels swerve drive mobile robot platform," in *2021 International Electronics Symposium (IES)*, 2021, pp. 669–674.

[5] Y. Jeong and S. Yim, "Model predictive control-based integrated path tracking and velocity control for autonomous vehicle with four-wheel independent steering and driving," *Electronics*, vol. 10, no. 22, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/22/2812

[6] L. Ge, Y. Zhao, S. Zhong, Z. Shan, F. Ma, Z. Han, and K. Guo, "Simultaneous stability and path following control for 4wis4wid autonomous vehicles based on computationally efficient offset free mpc," *International Journal of Control, Automation and Systems*, vol. 21, no. 9, pp. 2782–2796, 2023.

[7] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[8] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1714–1721.

[9] J. Yin, Z. Zhang, E. Theodorou, and P. Tsiotras, "Trajectory distribution control for model predictive path integral control using covariance steering," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1478–1484.

[10] T. Kim, G. Park, K. Kwak, J. Bae, and W. Lee, "Smooth model predictive path integral control without smoothing," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 406–10 413, 2022.

[11] L. Streichenberg, E. Trevisan, J. J. Chung, R. Siegwart, and J. Alonso-Mora, "Multi-agent path integral control for interaction-aware motion planning in urban canals," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1379–1385.

[12] G. Williams, B. Goldfain, P. Drews, K. Saigol, J. Rehg, and E. Theodorou, "Robust sampling based model predictive control with sparse objective information," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[13] M. S. Gandhi, B. Vlahov, J. Gibson, G. Williams, and E. A. Theodorou, "Robust model predictive path integral control: Analysis and performance guarantees," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1423–1430, 2021.

[14] F. Heetmeyer, M. Paluch, D. Bolliger, F. Bolli, X. Deng, E. Filicicchia, and T. Delbruck, "Rpgd: A small-batch parallel gradient descent optimizer with explorative resampling for nonlinear model predictive control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 3218–3224.

[15] E. Arruda, M. J. Mathew, M. Kopicki, M. Mistry, M. Azad, and J. L. Wyatt, "Uncertainty averse pushing with model predictive path integral control," in *International Conference on Humanoid Robotics*. IEEE, 2017, pp. 497–502.

[16] B. Liu, G. Jiang, F. Zhao, and X. Mei, "Collision-free motion generation based on stochastic optimization and composite signed distance field networks of articulated robot," *arXiv preprint arXiv:2306.04130*, 2023.

[17] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive sampling: Real-time behaviour synthesis with mujoco," *arXiv preprint arXiv:2212.00541*, 2022.

[18] L. Xu, Y. Yang, Q. Chen, F. Fu, B. Yang, and L. Yao, "Path tracking of a 4wis–4wid agricultural machinery based on variable look-ahead distance," *Applied Sciences*, vol. 12, no. 17, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/17/8651

[19] T.-H. S. Li, M.-H. Lee, C.-W. Lin, G.-H. Liou, and W.-C. Chen, "Design of autonomous and manual driving system for 4wis4wid vehicle," *IEEE Access*, vol. 4, pp. 2256–2271, 2016.

[20] W. Purbowaskito and C.-H. Hsu, "Kinematics model based motion control of a 4wis mobile robot with lyapunov stability," in *2020 12th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2020, pp. 68–73.

[21] M.-H. Lee and T.-H. S. Li, "Kinematics, dynamics and control design of 4wis4wid mobile robots," *The Journal of Engineering*, vol. 2015, no. 1, pp. 6–16, 2015. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/joe.2014.0241

[22] N. Hansen, X. Wang, and H. Su, "Temporal difference learning for model predictive control," 2022.

[23] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[24] I. Mezić and A. Banaszuk, "Comparison of systems with complex behavior," *Physica D: Nonlinear Phenomena*, vol. 197, no. 1, pp. 101–133, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278904002507

[25] P. J. SCHMID, "Dynamic mode decomposition of numerical and experimental data," *Journal of Fluid Mechanics*, vol. 656, p. 5–28, 2010.

[26] R. Chandra, *Parallel programming in OpenMP*. Morgan kaufmann, 2001.