NAMOUnc: Navigation among movable obstacles with decision making on uncertainty interval

Kai Zhang[1,2], Eric Lucet[1], Julien Alexandre dit Sandretto[2] and David Filliat[2]

***Abstract*— Navigation among movable obstacles (NAMO) is a fundamental task in various applications. Existing NAMO solutions usually assume that the robot operates in an ideal world with perfect observation and actions or a full knowledge on the environment. These assumptions limit their applicability on real robots and may result in risky actions. We propose a method (NAMOUnc) with consideration of multiple uncertainties, including observation noise, action failure, prediction uncertainty and uncertainty caused by partial observability. Based on the estimated uncertainties, the robot makes decision by comparing the time cost interval to reach the goal, then achieves the joint optimization on the time cost and success rate. We evaluate the proposed algorithm in both simulated and real environments and compare it with latest NAMO frameworks.**

I. Introduction

In real applications, robots take actions with partial and noisy observation on the environment, and a given action may cause unexpected effect due to uncertainties. If the robot has over confidence on the observation and action, it can lead to some suboptimal decisions, even dangerous actions resulting in catastrophic effects, like destroying objects in the workspace. It is therefore essential for the robot to recognize the limitation of its observations and actions, and make decisions with awareness of uncertainty and risks.

Recent works on task and motion planning incorporate uncertainty and update plans based on the observation and action uncertainties. For example [1], [2] take success rate (SR) into account in a manipulation task: if a grasp action fails, the planner updates its SR estimate, then replans for an action sequence with higher SR. Methods such as probabilistic symbolic planning [3] or Bayes optimization [4] plan with uncertainty but mainly focus on optimizing SR. They often neglect joint optimization with efficiency, which is crucial for navigation tasks.

Navigation among movable obstacles (NAMO) task is mainly a navigation task but the robot is able to manipulate movable obstacles (MO). Many existing solutions (e.g., [5], [6]) consider NAMO as a manipulation task, assuming manipulation is necessary to complete the task. However, the most common case is that the task can be finished without moving the MO. This oversight makes their probability-based optimization less useful, since when a bypass is possible, its SR being close to 1, it will be chosen even if it takes much more time. Besides, in partial observation (PO) condition, the invisible region in NAMO task is much larger than in the manipulation task. The common strategy of reducing this uncertainty in manipulation task by iteratively observing the occluded region [4], is inefficient or impractical for some NAMO tasks.

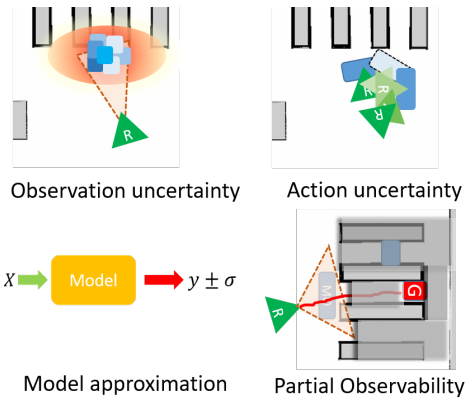

Fig. 1. Uncertainty sources in a NAMO task. Fours uncertainties are considered when the robot completes a NAMO task. The blue rectangle means the MO and the green triangle with R means the robot.

This paper presents NAMOUnc, a method for solving NAMO tasks by optimizing SR and running time in real scenarios. It considers several uncertainties (Fig. 1): (a) observation uncertainty from sensor noise, (b) model approximation uncertainty, (c) action uncertainty from imperfect controllers, and (d) blockage uncertainty from PO. NAMOUnc estimates these uncertainties as cost intervals and makes decisions based on their utility values, balancing removal and bypass strategies to achieve efficient and successful navigation.

In summary, our contribution includes: 1) A method to solve NAMO tasks with SR optimization and efficiency in partial observation conditions. 2) Four modules to systematically estimate and quantify the uncertainties described in Fig. 1 and a decision function to trade-off between SR and efficiency. 3) A novel method to estimate the uncertainty caused by PO in unexplored region, which can effectively reduce the navigation risk and improve the efficiency. 4) Experiments in simulated and real environments to demonstrate the effectiveness of our method.

II. Related Work

A. Navigation among movable obstacles

The NAMO task has been explored for a long time and recently proposed approaches to solve it include end-to-end [7] and hybrid methods [8], [5], [9]. The end-to-end methods are usually based on hierarchical reinforcement

Website: kai-zhang-er.github.io/namo-uncertainty/

[1] Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
[2] U2IS, ENSTA Paris, Institut Polytechnique de Paris, Palaiseau, France
kai.zhang@ip-paris.fr

learning [7], which produce high-level subgoals and low-level control parameters. The hybrid methods use machine learning either to generate subgoals [9] or help with the action sequence generation [5]. A recent detailed review can be found in [10].

All these methods are with the sole objective of completing the task regardless of the achieving cost. Our previous work [11] allows to choose a strategy based on estimated cost, jointly optimizing SR and efficiency. However, this method does not account for uncertainty, limiting its generalization capability.

B. Planning with uncertainty

Uncertainty in real environments leads to more challenges and methods are proposed to deal with different uncertainties such as observation uncertainty, action uncertainty and uncertainty linked to partial observation.

The *observation uncertainty* in task planning refers to the belief on object class and pose. The object classification uncertainty has been widely studied in computer vision field [12] and a confidence score is usually provided to quantify the uncertainty. The pose uncertainty results from the noisy sensors and it is typically modeled as Gaussian distribution and alleviated by repeated observation [13], [14].

The *action uncertainty* appeals to more attention. In the sampling based methods, the action uncertainty is described as a probabilistic transition matrix. This matrix can be obtained by approximation methods with frequent replanning [14] or learning methods that gather information from demonstration or datasets [3]. With the transition matrix, the most likely successful plan is selected and executed. Rather than take it as open-loop plan, some methods [2] update the transition probability after observing the action effect and adjust the plan iteratively to achieve the task.

Regarding *uncertainty from partial observation*, most methods [5], [4], [15] constrain actions to the sensor's field of view. In a manipulation task, [4], [14] characterize the probability of unseen objects behind the visible ones and use it in the planning to perform exploration actions when necessary. For NAMO, RNAMO [15] simply proposes that the robot pushes all obstacles blocking the path in the invisible area. If the obstacles are not movable, an internal map is updated and a detour is planned to the goal. In larger workspace, LaMB [5] uses backward reasoning to eliminate environmental invisibility for task planning. However, this approach is limited to small workspace as it explores all invisible regions which is impractical in large environments. To minimize the cost, we propose a method to quantify the possible cost to go through the invisible region and find an optimized trade-off between exploration (bypass in unknown area) and exploitation (move object in visible area).

III. Navigation among movable obstacles

In a NAMO task, shown in Fig 2, a robot needs to navigate to a goal while avoiding obstacles. With the environment map, the first step is to plan and follow the shortest path. If the robot meets an MO blocking this path, it can choose to bypass it or clear the path by removing it. As described in our previous work [11], we first estimates the bypass and removal cost before making decision. The bypass cost is calculated based on a detour trajectory. The removal cost is computed in two steps: predicting the stock region for the MO, then estimating the time to more the MO to this region. According to the decision, the motion planner outputs the control parameters for the robot to execute the task.

The uncertainty in various modules can cause task failure. In MO detection and localization, the observation uncertainty, including recognition and pose estimation uncertainty, may result in collision or the failure of removal action. For the cost estimation of bypass and removal, the model uncertainty may produce an incorrect estimate. If the robot chooses to remove the MO, the action may fail due to the insufficient knowledge or the discrepancy between estimated and actual action effect. Additionally, navigating in partially observed environments introduces inherent uncertainty, creating a gap between expectation and reality.

IV. Uncertainty estimation method

We detail the four kind of uncertainties considered in the NAMO tasks and the methods to estimate them. We use time intervals, represented as $[\ ]$, to represent the effect of uncertainty on the decision-making module.

A. Observation uncertainty

While navigating the environment, the robot should localize the MOs and detect whether the planned path is blocked. Due to the sensors' noise in object detection and localization error of the robot, the estimated MO pose is subject to uncertainty. However, multiple observation can refine the result, and given a specified confidence interval, a belief region can be calculated for path blockage determination.

The obstacle pose is the robot pose plus the offset from robot frame. Assuming the robot pose from the localization algorithm is $X_r = (x_r, y_r, \theta_r)$ with covariance Σ_r, and the relative distance and angle of the i-th MO measured by the depth camera $Y^i = (d^i, \phi^i)$ with covariance Σ_Y, the obstacle pose X^i_{MO} can be obtained by: $X^i_{MO} = (x_r + d^i cos(\theta_r + \phi^i), y_r + d^i sin(\theta_r + \phi^i))^T$ and the covariance matrix Σ_{MO} by:

$$\Sigma^i_{MO} = J_r \Sigma_r J_r^T + J_y \Sigma_Y J_y^T; \quad J_r = \frac{\partial X^i_{MO}}{\partial X_r}; \quad J_y = \frac{\partial X^i_{MO}}{\partial Y^i}$$

When multiple observations on the same MO are received, a Kalman filter [16] is applied to fuse the repeated observation and obtain the estimated MO pose and its covariance.

Given a confidence score T_{conf} (we use $T_{conf} = 95\%$), the belief region of the MO pose is represented as an ellipse computed from the covariance matrix. We add the size of the MO, expressed as its radius, to obtain the ellipse region where a path would lead to collision. In this case, the robot stops and choose the best strategy as described in Sec. IV-E.

B. Bypass cost model uncertainty

Bypass cost, i.e., the estimated travel time to reach the goal when the robot follows the planned detour, varies according

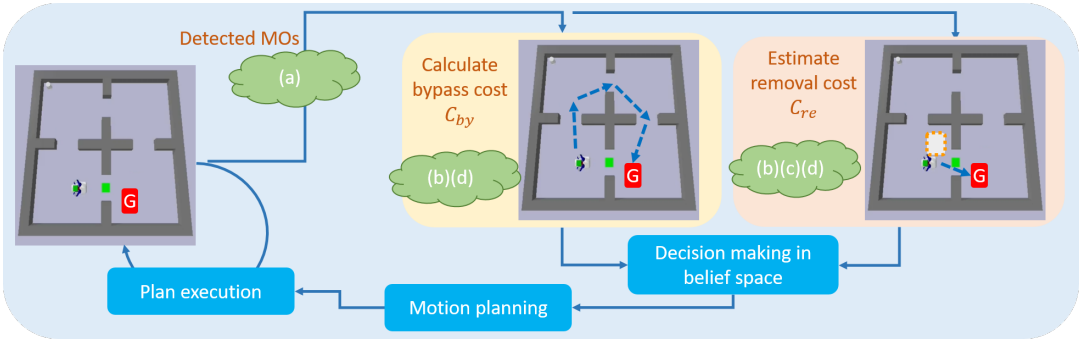

Fig. 2. The pipeline of NAMO method. The green cloud means the uncertainty involved in the module: (a) Observation uncertainty; (b) Model uncertainty; (c) Action uncertainty; (d) Blockage uncertainty caused by partial observability.

to the trajectory and moving speed. The model that predicts navigation time for the trajectory has uncertainty that we represent as an interval.

To plan the detour, the MO and its uncertainty region (ellipse with 95% confidence described in sec IV-A) is temporarily added to the obstacle map, then the shortest path planner searches an alternative path to bypass the MO. If no path is found, the bypass time cost is set to $C^{by}_{nav} = Inf$. If there is, we need to estimate the navigation time from trajectory features. While the average speed was simply used in [11], we use a Gaussian linear regressor (GLR) [17] for better prediction and uncertainty evaluation (see Sec. V-C).

Indeed, as rotation takes more time than following straight lines, we need to take the orientation change into account during bypass time estimation. In addition to the trajectory length F_l, we therefore calculate the trajectory smoothness F_s and variance of direction change F_v as features to estimate the navigation time. Assuming a trajectory consists of N waypoints $pt_i, i = 0, 1, .., N$, each pt_i characterized by position p_i and orientation α_i, smoothness and variance are calculated using: $F_s = \frac{\sum_{i=1}^{N} |\alpha_i - \alpha_{i-1}|}{N-1}$, $F_v = var(|\alpha_i - \alpha_{i-1}|)$. A trajectory is therefore characterized by $X = \{F_l, F_s, F_v\}$.

The bypass time cost and variance are predicted by GLR: $T_{by}, \sigma_{by} = GLR(X)$. The navigation time interval with T_{conf} (95%) confidence is : $[C^{by}_{nav}] = [T_{by} - 2\sigma_{by}, T_{by} + 2\sigma_{by}]$

To train the regressor, we collect a set of trajectories by controlling the robot to navigate in a warehouse environment. The pose and time stamp are recorded along these trajectories, and we create a large and varied dataset for the regressor by sampling random start and goal points in these trajectories and computing the corresponding features and duration.

C. Removal action uncertainty

The action uncertainty relates to the uncertain outcome of loading a MO for displacement, which can be either success or failure. Similar to [4], we model the success rate (SR) of the action and the uncertainty of its estimation after t trials, p^t_a, using a Beta distribution.

We start by an initial estimate of the SR and update it during robot operation to reflect any change in the robot behavior. To obtain the initial SR, p^0_a, we control the robot to load the MO in several trials and record the action results. Assuming there are α successful trials and β failure cases, the initial knowledge on the trial results can be described as $p^0_a \sim Beta(\alpha, \beta)$. During operation, when new trials are performed, the updated posterior is $p^t_a \sim Beta(\alpha + s, \beta + f)$ where s and f are the number of successful and failed object movement respectively. With a confidence score T_{conf} ($T_{conf} = 95\%$ in our experiments), we can obtain the SR interval $p^t_a \in [Beta.ppf(0.025), Beta.ppf(0.975)]$ where the $Beta.ppf$ is the point percent function to obtain the confidence interval of a distribution given a confidence score.

For a given SR, the expected removal cost C_{MO} is

$$C_{MO} = T_{MO} \sum_{i=1}^{M} i p^t_a (1 - p^t_a)^{i-1} + (M T_{MO} + C_{by})(1 - p^t_a)^M$$

where M is the maximum times that the robot will try when it fails to load a MO ($M = 5$ in our experiments), C_{by} is the obstacle bypass cost, and T_{MO} is the removal cost of a MO that can be estimated by the method proposed in [11], where a stock region predictor and a regressor are employed to predict the placing pose and the removal time.

We compute the interval of the expected removal cost $[C_{MO}]$ by using this formula with the minimum and maximum p^t_a of the SR interval.

D. Blockage uncertainty

Blockage uncertainty comes from the partial observation condition. It is the blocking probability of the robot by some unseen MOs in unexplored region. It is related to the passage width and robot size as it is more risky if the planned navigation path passes a narrow passage than the open space.

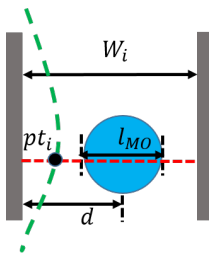

Fig. 3. Blocking case. The blue circle represents the MO in a corridor with width W_i. The green dash curve is the planned trajectory of the robot while the red dash line is the traversal line at waypoint pt_i

To calculate the blockage probability of a trajectory T, we take T as a set of way points $pt_i, i = 1,2,...,N$ and compute the blockage probability of each way point.

We model the blockage probability based on several assumptions (shown in Fig. 3): (i) The width of the passage W_i at pt_i, (ii) The radius of the robot r, (iii) The diameter of the MO, l_{MO}, modeled as a Gaussian distribution $G(\mu, \sigma)$ where μ is the average radius of the MOs and σ is the standard deviation. (iv) The distance between the MO center and the wall d, modeled as a uniform distribution $d \sim U(\frac{l_{MO}}{2}, W_i - \frac{l_{MO}}{2})$, which leads to $p(d|l_{MO}, W_i) = \frac{1}{W_i - l_{MO}}$.

Given d, l_{MO}, r and W_i, the blockage probability $p(b|d, l_{MO}, W_i, r)$ is deterministic:

$$p(b|d, l_{MO}, W_i, r) = \begin{cases} 1, & (2r > d - \frac{l_{MO}}{2}) \wedge \ (2r > W_i - d - \frac{l_{MO}}{2}) \\ 0, & otherwise \end{cases}$$

To obtain the blockage probability conditioned only on the passage width and the robot size $p(b|W_i, r)$, we eliminate the dependence on d, l_{MO} through marginalization:

$$p(b|l_{MO}, W_i, r) = \int_{\frac{l_{MO}}{2}}^{W_i - \frac{l_{MO}}{2}} p(b|d, l_{MO}, W_i, r) p(d|l_{MO}, W_i, r)\, d_d$$

The blockage probability in a corridor is then:

$$p(b|W_i, r) = \int p(b|l_{MO}, W_i, r) p(l_{MO})\, d_{l_{MO}}$$

Considering that l_{MO} satisfies a Gaussian distribution and $p(b|l_{MO}, W_i, r)$ is piecewise constant, we approximate this integral by using a sampling method.

The previous $p(b|W_i, r)$ is calculated with the assumption of a MO being on the line perpendicular to the corridor passing through pt_i (the red dash line in Fig. 3). Because W_i can be obtained from pt_i using a ray casting algorithm, and both are independent of r, the probability of blockage at pt_i can be expressed as $p(b|pt_i, r) = p(b|W_i, r)$.

Assuming the MO is uniformly distributed in the space with a free area A, the probability that the MO is in the traversal line at pt_i is $p(pt_i) = \frac{W_i K}{A}$. Here K is a parameter characterising the obstacle appearance probability.

Therefore, the probability that the robot is blocked at pt_i is $p(b|pt_i, r) \times p(pt_i)$. Then, for a trajectory T, the blockage probability $p(b|T, r)$ can be computed by:

$$p(b|T, r) = 1 - \prod_{pt_i}^{T} (1 - p(b|pt_i, r) \times \frac{W_i K}{A})$$

The estimated cost of blockage when passing the invisible region is then : $[C_{blocked}] = p(b|T, r) \times [C_{MO}]$

E. Decision making with uncertainty interval

The decision making module aims to compare the cost of bypass and removal, and choose the one with smaller cost.

With the uncertainties considered, the final cost of each option can be calculated by $[C_{by}] = [C^{by}_{nav}] + [C^{by}_{blocked}]$; $[C_{re}] = [C_{MO}] + [C^{re}_{nav}] + [C^{re}_{blocked}]$. where $[C^{by}_{blocked}]$ and $[C^{re}_{blocked}]$ are the blockage costs of bypass and removal trajectories.

For the decision making between the cost intervals, we apply the Laplace criterion, described in [18], to compute the average utility of the consequences of each option. Assuming the cost satisfies the uniform distribution, the utility function can be expressed as:

$$U = \int_{min([C])}^{max([C])} xp(x)\, dx = \frac{max([C]) + min([C])}{2}$$

where [C] is either $[C_{re}]$ or $[C_{by}]$ to calculate the utility while x and $p(x)$ are samples in the cost interval and its probability. Finally, the option with a smaller U is chosen as the navigation strategy.

V. Simulation Experiments

We first conduct individual modules evaluation in simulation, and then compare our method with the state of the arts before demonstrating a real robot application.

A. Simulation environment

We implement our method in two simulated environments, a simple room and a large warehouse, as shown in Fig 4. There is one MO in the room while multiple MOs can be in the blue regions in the warehouse. A wheeled mobile robot with an arm needs to complete navigation tasks. The environment map (including only the static obstacles) is generated using GMapping [19] and provided to the robot as prior knowledge. A LiDAR and a stereo camera are used to localize the robot and detect MOs respectively.

Fig. 4. Simulation environments.Two environments are used including a simple room and a complex warehouse. The blue regions are possible places for the MOs and the red region is the goal.

B. Implementation details

The room environment is built on PyBullet [20] while the warehouse is based on Gazebo [21] and ROS Noetic [22] with Movebase as the navigation framework. The robot detects the MOs by Aruco Marker [23] to reduce the class uncertainty. When a MO blocks the planned path, the decision module chooses a suitable avoidance strategy between bypass and removal. All the experiments are implemented in Python with PyTorch [24]. We use an Intel i7-12700H CPU with 16G memory for the quantitative results.

C. Bypass time regression results

To demonstrate the improved bypass time prediction, we compare the applied GLR predictor with the average speed method from [11] and a trapezoid method that considers acceleration and deceleration. We collect a dataset including 1500 trajectory segments as training and 600 as testing. The

average speed method calculates the speed in the training dataset, then applies it in the test set. For the GLR method, a regressor is fitted on the training set to predict the test set. We calculate the absolute error between the predicted and actual navigation time. The result in Fig. 5 shows that the applied GLR method outputs more accurate results with the lowest median absolute error (1.59s), compared to the average speed (3.43s) and trapezoid methods (3.31s). Additionally, the GLR method is more stable, with an interquartile range (IQR) of 0.69s, versus 1.35s and 1.91s for the other methods.

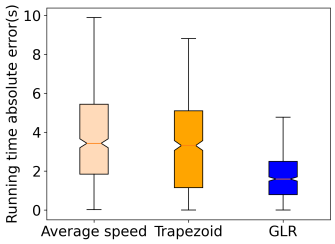

Fig. 5. Boxplot of the absolute error of prediction results for the three bypass time prediction methods. The box represents the interquartile range (IQR), with the lower and upper edges indicating the 25th (Q1) and 75th (Q3) percentiles, respectively. The notch and orange line in the box marks the median value of the absolute error. Whiskers extend to the smallest and largest values within 1.5 times the IQR from Q1 and Q3.

D. Action uncertainty module evaluation

To evaluate the effectiveness of modeling action uncertainty, we compare the task completion time with (w/) and without (w/o) the action uncertainty module in two cases: one with easy MOs (90% loading SR) and one with hard MOs (20% SR). We test the methods in three setups, ABC, AB, BC. ABC (resp AB and BC) indicates three MOs are in regions A, B and C (resp. 2 at AB and BC) in Fig. 4.

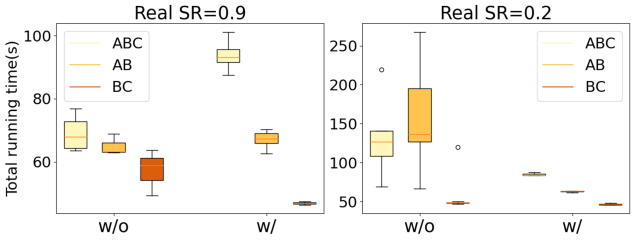

Fig. 6. Task completion time comparison on methods with/without considering action uncertainty in three environments, ABC, AB, BC. The left figure shows the case with easy MO (high SR) while the right one shows the hard MO (low SR).

The results in Fig 6 show that with easy MOs (SR=0.9), the w/o method takes less time in planning and making decision as it removes the MO at B directly. Conversely, the method w/ has the robot bypass B first to check MOs in A or C. If A and C are also blocked, the robot returns and removes the MO in B, requiring more planning and navigation time.

When MOs are hard to manipulate (SR=0.2), the method w/ bypass all MOs due to the high potential cost of removal action, leading to faster navigation compared to w/o method that repeatedly attempts to remove MOs regardless of the cost. As for the stability, the w/ method demonstrates much lower IQR with 2.53s for easy MOs and 1.35s for hard MOs, compared to the w/o method's 6.16s and 34.14s, respectively.

E. Ablation study on bias between estimated SR and real SR

To obtain the initial SR value, we conduct prior experiments, as explained in Sec. IV-C. However, the estimated and actual SR may differ, especially when assuming all MOs share the same SR. Since SR relates to the estimated removal cost and affects the navigation strategy, we analyze the impact of errors in estimating the SR. The results on the cases with and without estimation bias in environment ABC are shown in Table I. All the times in the table are the average of 5 trials. In method with action uncertainty (w/), an unbiased SR estimation gives the best navigation strategy with minimal time. Even with biased estimation, the method w/ still outperforms the method w/o, proving the effectiveness of the proposed action uncertainty module.

TABLE I

AVERAGE RUNNING TIME OF METHODS WITH UNBIASED/BIASED ESTIMATION ON SR. THE CELL MARKED BOLD MEANS BETTER RESULT.

	Estimated SR	Real SR	w/o	w/
unbiased	0,90	0,90	**68,99±4,42**	93,80±4,53
	0,50	0,50	98,31±21,81	**94,19±12,86**
	0,20	0,20	164,15±80,89	**85,14±1,92**
	Avg		110,48	**91,04**
biased	0,90	0,20	**164,15±80,89**	168,19±39,50
		0,50	**98,31±21,81**	109,24±15,91
	0,50	0,20	164,15±80,89	**113,46±22,17**
		0,90	**68,99±4,42**	91,03±6,95
	0,20	0,50	98,31±21,81	**85,57±2,31**
		0,90	**68,99±4,42**	84,45±2,42
	Avg		110,48	**108,66**

F. Blockage uncertainty module evaluation

To compare the impact of introducing the blockage uncertainty module, we design two environments AB and ABE for evaluation (with obstacles at the corresponding positions shown in Fig. 4). Environment AB demonstrates the difference of navigation strategy due to blockage uncertainty while ABE illustrates the advantage of the blockage uncertainty module when an unexpected MO appears on the detour.

The evaluation results in Table II report the mean runtime of 5 trials. In environment AB, without considering the blockage uncertainty (w/o), the robot bypasses all the MOs. In contrast, the w/ method chooses to remove MO in region B considering the potential blockage risk of the detour in narrow passage. In AB where no surprising MO appears, the bypass takes less time than the removal. However, in ABE, where an unexpected MO blocks the detour, the method w/o bypasses the MO in region B, then bypass E (failing to find a suitable stock region for moving E), finally it returns to remove the MO at B, taking much longer time than the method w/ that removes B initially. From the overall performance, the method w/ is more efficient comparing to the method w/o.

TABLE II

AVERAGE RUNNING TIME OF METHOD WITH/WITHOUT BLOCKAGE UNCERTAINTY IN TWO ENVIRONMENTS

	Methods	
Env	w/o	w/
AB	**67,04±2,54**	77,77±2,63
ABE	141,02±16,81	**90,08±2,25**
Overall	104,03	**83,92**

G. Overall comparison

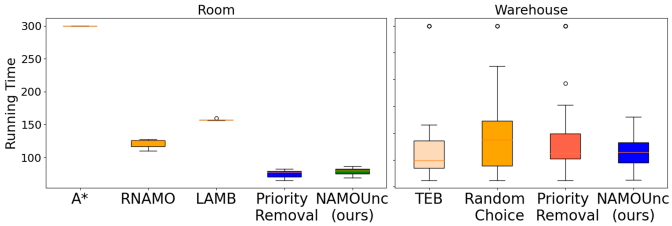

Fig. 7. Running time in two simulation environments, the room (left) and the warehouse (right). The outliers (circle around 300 in the figures) represent the failure cases. The orange line in the box marks the median value.

We evaluate the overall performance in the two simulation environments (Fig. 4), with configurations of ABC, AB, BC, ABE, ABD, BCE in the warehouse. The starting points and goal points are randomly selected to create different navigation tasks. Then, we compare our method with some baseline methods, priority bypass [25], [26], priority removal, random choice method and LaMB [5].

The priority bypass methods including TEB [25] and A* [26], bypass all the obstacles but fail if there is no alternative way to the goal. The priority removal method refers to a category of NAMO methods [6], [27], which removes the MOs that block the path without considering the removal cost. The random choice method chooses to remove or bypass with a probability (0.5 in our experiments). The LaMB method [5] is one of the latest methods considering the partial observation constraints in NAMO tasks but limited to small scale environments.

We record the average running time (ART) and success rate SR of each method. Tasks are marked as failed if the goal is not reached within 300 seconds. Results for ART are shown in Fig. 7. In the room environment, the only path to the goal is blocked by a MO. Therefore, the bypass method (A*) always fails. The NAMOUnc and priority removal methods complete the task more quickly, with the priority removal method slightly faster (77.68s vs. 79.56s) since it takes less time to plan bypass and make decision. In the warehouse environment, the NAMOUnc and the TEB methods finish the task with comparable time cost but NAMOUnc has no failures.

VI. REAL EXPERIMENTS

A. Environment description

To evaluate the performance of our method in real applications, we use a real Jackal robot in a small warehouse-like environment. As shown in Fig. 8, there are maximum 3 MOs and the robot should navigate to a goal G. It is equipped with a LiDAR and a realsense camera to observe the environment, and an arm to lift the MO.

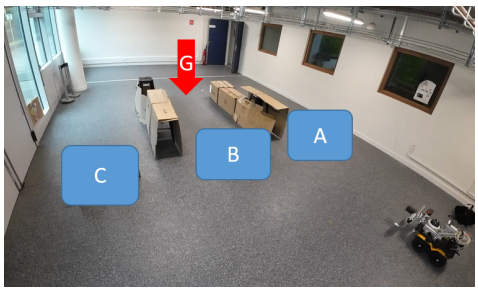

Fig. 8. Real environment setup. The blue regions marked as A, B, C are possible positions of MOs. The red arrow with G is the goal.

B. Experiment results

We randomly pick goal points to create different NAMO tasks and record the running time with different MO setups, including environments ABC, AB and BC. The real SR is set to 100% and quantitative results are shown in Table III, where each cell indicates the average running time and corresponding standard deviation on 5 trials. Although the proposed method does not win the first place in each environment, from average running time, it achieves the best overall performance on 3 setups. This shows that the proposed method achieves a good trade-off between completeness and efficiency in the search for a solution.

TABLE III

OVERALL PERFORMANCE COMPARISON AMONG DIFFERENT METHODS. THE COLUMNS MARKED BOLD REPRESENT THE BEST RESULT

	TEB[25]	Priority Removal	Random Choice	NAMOUnc (Ours)
ABC	N/A	**96.33±9.01**	115.34±14.58	137.69±13.19
BC	**46.66±4.56**	94.78±5.08	72.73±19.94	50.41±2.62
AB	**61.11±4.90**	97.94±5.72	107.44±22.40	72.56±2.09
Overall		96.35	98.5	**86.88**

VII. CONCLUSION AND FUTURE WORK

We have presented a NAMO framework capable of planning the task and motion under four kinds of uncertainty: observation, action, model and blockage uncertainty. Our planner jointly optimizes SR and running time. Experimental results in both simulation and real environments demonstrate its ability to balance these objectives, suggesting potential extensions to optimize additional objectives like energy and safety.

ACKNOWLEDGMENT

This work was carried out as part of the OTPaaS project. This project received funding from the French government as part of the "Cloud Acceleration Strategy" plan.

REFERENCES

[1] E. Safronov, M. Colledanchise, and L. Natale, "Task planning with belief behavior trees," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6870–6877.

[2] T. Pan, A. M. Wells, R. Shome, and L. E. Kavraki, "Failure is an option: task and motion planning with failing executions," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 1947–1953.

[3] T. Silver, R. Chitnis, J. Tenenbaum, L. P. Kaelbling, and T. Lozano-Pérez, "Learning symbolic operators for task and motion planning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3182–3189.

[4] A. Curtis, G. Matheos, N. Gothoskar, V. Mansinghka, J. Tenenbaum, T. Lozano-Pérez, and L. P. Kaelbling, "Partially observable task and motion planning with uncertainty and risk awareness," *arXiv preprint arXiv:2403.10454*, 2024.

[5] J. Muguira-Iturralde, A. Curtis, Y. Du, L. P. Kaelbling, and T. Lozano-Pérez, "Visibility-aware navigation among movable obstacles," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 083–10 089.

[6] K. Ellis, H. Zhang, D. Stoyanov, and D. Kanoulas, "Navigation among movable obstacles with object localization using photorealistic simulation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 1711–1716.

[7] C. Li, F. Xia, R. Martin-Martin, and S. Savarese, "Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators," in *Conference on Robot Learning*. PMLR, 2020, pp. 603–616.

[8] B. Kim, Z. Wang, L. P. Kaelbling, and T. Lozano-Pérez, "Learning to guide task and motion planning using score-space representation," *The International Journal of Robotics Research*, vol. 38, no. 7, pp. 793–812, 2019.

[9] F. Xia, C. Li, R. Martín-Martín, O. Litany, A. Toshev, and S. Savarese, "Relmogen: Integrating motion generation in reinforcement learning for mobile manipulation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4583–4590.

[10] K. Zhang, E. Lucet, J. Alexandre Dit Sandretto, S. Kchir, and D. Filliat, "Task and motion planning methods: applications and limitations," in *ICINCO 2022 - 19th International Conference on Informatics in Control, Automation and Robotics*, ser. Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics - ICINCO. Lisbonne, Portugal: SCITEPRESS - Science and Technology Publications, July 2022, pp. 476–483. [Online]. Available: https://hal.science/hal-03744923

[11] K. Zhang, E. Lucet, J. Alexandre dit Sandretto, and D. Filliat, "Navigation among movable obstacles using machine learning based total time cost optimization," 2023.

[12] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A review and comparative study on probabilistic object detection in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 9961–9980, 2021.

[13] L. P. Kaelbling and T. Lozano-Pérez, "Integrated robot task and motion planning in the now," 2012.

[14] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Online replanning in belief space for partially observable task and motion problems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5678–5684.

[15] H.-n. Wu, M. Levihn, and M. Stilman, "Navigation among movable obstacles in unknown environments," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 1433–1438.

[16] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.

[17] C. Williams and C. Rasmussen, "Gaussian processes for regression," *Advances in neural information processing systems*, vol. 8, 1995.

[18] T. Denoeux, "Decision-making with belief functions: A review," *International Journal of Approximate Reasoning*, vol. 109, pp. 87–110, 2019.

[19] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[20] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.

[21] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ IROS*, vol. 3. IEEE, pp. 2149–2154.

[22] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system." [Online]. Available: https://www.ros.org

[23] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Speeded up detection of squared fiducial markers," *Image and vision Computing*, vol. 76, pp. 38–47, 2018.

[24] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[25] C. Rösmann, F. Hoffmann, and T. Bertram, "Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control," in *2015 european control conference (ECC)*. IEEE, 2015, pp. 3352–3357.

[26] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[27] M. Wang, R. Luo, A. Ö. Önol, and T. Padir, "Affordance-based mobile robot navigation among movable obstacles," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 2734–2740.