Extending Task and Motion Planning with Feasibility Prediction: Towards Multi-Robot Manipulation Planning of Realistic Objects

Smail Ait Bouhsain, Rachid Alami, Thierry Simeon

Extending Task and Motion Planning with Feasibility Prediction: Towards Multi-Robot Manipulation Planning of Realistic Objects

Smail Ait Bouhsain[1], Rachid Alami[1] and Thierry Siméon[1]

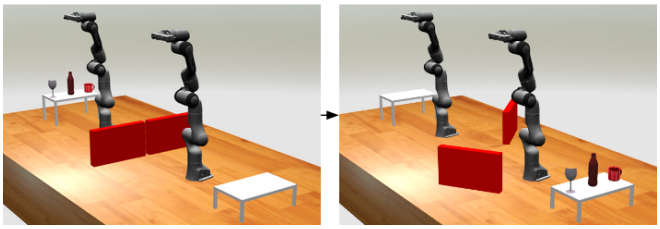
***Abstract*— The hybrid discrete/continuous nature of task and motion planning (TAMP) results often in a combinatorial explosion. This challenge is even more pronounced in multi-robot TAMP problems due to the increase in dimensionality of the action space. Previous works use action feasibility prediction as a heuristic to accelerate TAMP. However, these methods are limited to box-shaped objects and specific single or dual robot settings. In this paper, we expand on our previous work on action and grasp feasibility prediction [1] by extending its use to complex-shaped objects and multi-robot systems. Also, we propose a feasibility-enabled multi-robot TAMP algorithm capable of tackling complex multi-robot manipulation problems. We demonstrate the performance of our method compared to a non feasibility-informed baseline, and show its ability to handle TAMP problems requiring the collaboration of multiple robots.**

I. Introduction

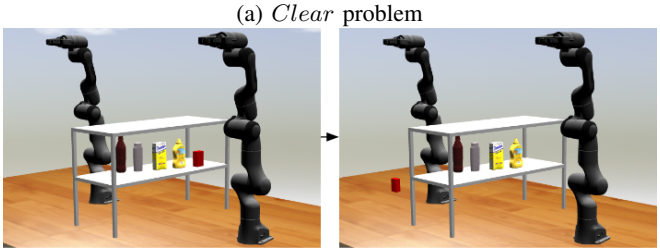
Task and motion planning (TAMP, see e.g survey [2]) in robotics involves finding a sequence of steps a robot should take to achieve a goal, along with their corresponding motions. It mixes discrete symbolic planning and continuous geometric planning. This combination results in a high combinatorial complexity making the search for a geometrically feasible task plan tedious, time consuming and, in some cases, unsolvable in a reasonable amount of time. These challenges do not only come from the combinatorial complexity of the search, but also the high time cost of calling geometric planners to verify the feasibility, particularly infeasibility, of actions and plan their motions.

Recent works [3]–[6] leverage learning methods in order to tackle this shortcoming of geometric planners. They propose to learn to predict the feasibility of actions without the need for querying geometric planning. These feasibility predictions can then be used as heuristics during task and motion planning. In a previous work [1], we propose the **AGFP-Net** neural network, which predicts the feasibility of pick and place actions in 3D environments, as well as the feasibility of subsets of grasps. This model is integrated in a feasibility-informed TAMP algorithm which uses feasibility predictions as a heuristic to accelerate planning. The method proposed in [1] is however limited to box-shaped objects and obstacles. Also, it is limited to single-robot TAMP problems and is not able to handle multi-robot settings.

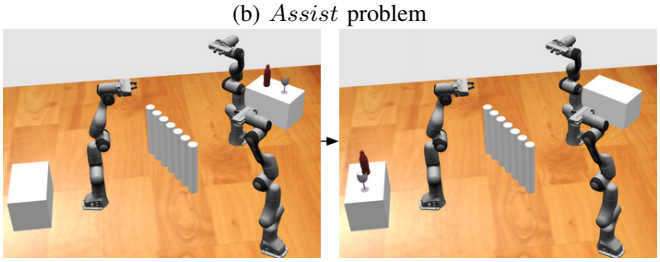
In this work, we tackle these limitations by extending the use of **AGFP-Net** to complex-shaped objects, allowing efficient action and grasp feasibility predictions in realistic environments containing real-life objects. We also develop a framework for feasibility prediction in multi-robot TAMP problems, and introduce a new way of estimating the feasibility of composite actions involving more than one robot. Moreover, we propose a feasibility-informed multi-robot TAMP algorithm capable of solving single and multi-robot problems such as the ones shown in Figure 1, while taking advantage of feasibility prediction to notably accelerate the planning process.



(a) *Clear* problem



(b) *Assist* problem



(c) *Middleman* problem

Fig. 1: A visualization of the initial and goal states of 3 of the multi-robot TAMP problems solved by our method.

II. Related Work

A. Classical TAMP

Early TAMP research [7]–[15] focused on the geometric aspect, approaching the problem as a multi-modal motion planning problem, which involves planning among multiple motion modes. Many current TAMP techniques [16]–[23] integrate a task planner with a geometric planner, relying on geometric backtracking to connect the two. These approaches, nevertheless, face challenges due to the combina-

[1]LAAS-CNRS, Toulouse, France, {saitbouhsa, alami, simeon}@laas.fr

This work is partially supported by the EU-funded project euROBIN under grant agreement no. 101070596 (https://www.eurobin-project.eu/)

torics of the hybrid discrete/continuous search. and the heavy time consumption of geometric planning, as verifying the feasibility of symbolic plans still necessitates a large number of costly calls to the geometric planner.

B. Learning for TAMP

In recent years, learning methods became increasingly popular in TAMP [3]–[6], [24]–[34], [1]. Many works aim at providing a learned heuristic to a TAMP planner, such as evaluating action affordances [29], leveraging experience to propose promising actions [26], [27], or providing fast geometric feedback to the task planner thanks to action feasibility prediction [3]–[6]. These methods are however limited to tabletop environments. In [1], [33], we propose a learning approach for predicting action and grasp feasibility in 3D environments. As most feasibility prediction methods, ours is limited to box-shaped objects. Wells et al. [3] propose to use SVMs to predict the feasibility of pick and place actions on complex-shaped objects. These objects are represented using hand-designed features, which makes the generalizability to various shapes difficult. Park et al. [35] propose a neural network for predicting the feasibility of grasp modes in scenes containing one complex-shaped object and a number of obstacles. However, this method is object-centric, meaning that the proposed model needs to be retrained for each new object shape. In this paper, we extend our action and grasp type feasibility prediction neural network [1] to complex-shaped objects, while maintaining generalizability to 3D environments containing multiple objects of various shapes.

C. Multi-Robot TAMP

Multi-robot TAMP suffers from a higher combinatorial complexity due to the increased number of robots, and thus the increased dimensionality of the action space [36]. Previous works [16], [37]–[41] propose various approaches for task and motion planning in multi-robot settings such as graph-based and MDP-based techniques. These methods suffer however from a combinatorial explosion as the number of robots increases, resulting in a notable increase in the already high number of calls to the geometric planner. Also, in most multi-robot TAMP research, the focus is on coordinated motion planning (e.g handover tasks). In this paper, we propose to accelerate multi-robot TAMP using robot-centric feasibility predictions, allowing the planner to mitigate the combinatorial explosion due to the presence of multiple robots. We focus on sequential motion planning, rather than coordinated motion planning, in which objects are passed between two robots using a pair of pick-place actions. Driess et al. [4], [28] train a neural network to predict action feasibility on box-shaped objects in tabletop environments using two robotic arms. This learning approach is limited, nevertheless, to the multi-robot setting it is trained on, requiring a new training if the number of robots increases, or the robots' placements change. Our proposed feasibility prediction framework does not depend on these parameters, since it queries the neural network for each robot individually. Park et al. [35] propose a similar method, but focus on environments containing one movable object only.

III. Problem Description

We tackle manipulation problems where the goal is to rearrange a set of movable objects in a 3D environment, comprising n_R robot arms with fixed bases, n_{ss} stable support surfaces, n_{obs} fixed obstacles and n_O movable objects.

We define the state of the environment as the configuration of each robot together with the support surface and pose of all movable objects. The configuration of a robot r at state s is denoted $s(r)$, whilst the configuration of an object O is denoted $s(O)$. A transition between two states s and s' is defined as an action a which involves a specific robot r picking an object O from its configuration in s, and placing it at a new configuration $\mathbf{q}$, which becomes the configuration of O in s'. The action a can be explicitly represented as:

$$a = Move(r, O, s(O) \rightarrow \mathbf{q}) \tag{1}$$

The solution to the TAMP problem is a sequence of actions τ and their corresponding motions Π, which brings the environment from the initial state s_0 to a goal state s_{goal}. We define 3 types of actions at the symbolic level. The first is a $Goal$ action, which moves an object to its goal state. The second is a $Temp$ action, which places an object at a temporary placement. The third is a $Pass$ action, which aims at placing an object inside the intersection between two robots' workspaces, so that the second robot can reach it. We consider the workspace of a robot r, denoted $W(r)$, as a sphere centered at the first joint of the robot, with a radius equal to its reach[1].

IV. Feasibility prediction

A. Neural Network

In [1], we propose a learning approach based on the Action and Grasp Feasibility Prediction neural NETwork (**AGFP-Net**). Given a representation of the current state of the environment and the object of interest, it simultaneously predicts the probability of feasibility of a $Pick$ or $Place$ action, but also the feasibility of 6 grasp types. Each grasp type is a set of grasps related to the side from which the object is approached by the robot. The 3D environment is represented using 5 depth images corresponding to different views of the scene, which show all the objects in the workspace of the robot. Also, all objects poses are shown in the frame of the robot. Note that these depth images are constructed internally during planning, and are not obtained from depth cameras. The object of interest, on the hand, is represented using a mask over each depth image, showing the object only at the pose it should picked or placed at.

AGFP-Net is trained on a fully synthetic dataset, comprised of randomly generated scenes containing a single robot, 2 box-shaped objects and up the 4 support surfaces. The dimensions and poses of these objects are sampled

[1]Note that this simple model of $W(r)$ is generalizable to more precise workspace representations, provided that mapping the intersection between two workspaces is possible.

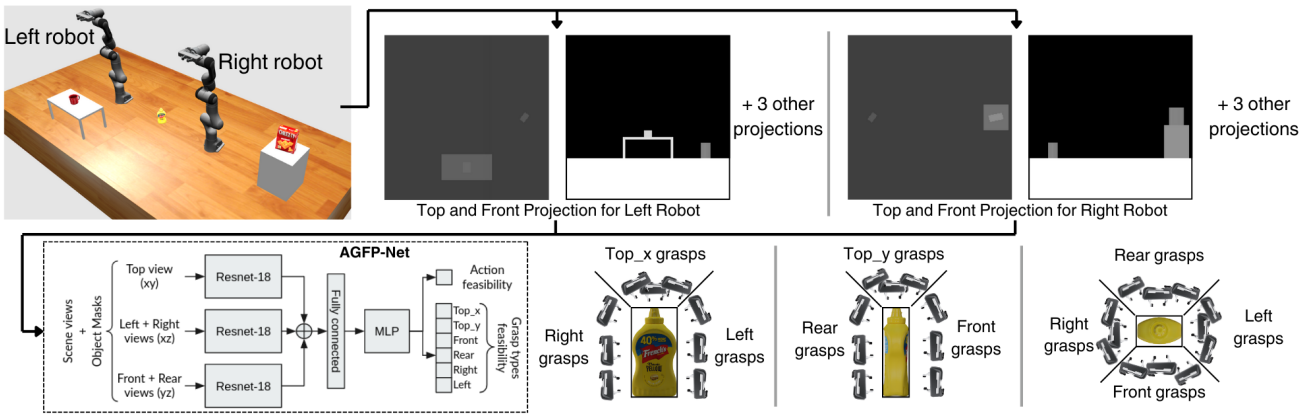


Fig. 2: Visualization of the feasibility prediction pipeline on an example 2-robot scene. **(Top)** Scene depth images are constructed for each robot such that only the objects in the workspace of the robot are shown. Objects present is both robots workspaces are visible in both scene projections (yellow mustard bottle). **(bottom)** Scene projections and object masks are given to **AGFP-Net** to obtain the action feasibility and the feasibility of 6 grasp types, illustrated on an example object.

randomly within fixed bounds. Our results in [1] show a good generalization to new environments with a higher number of objects, obstacles and support surfaces. Nonetheless, the neural network is by design limited to a single robot, and it can be used on box-shaped objects only. We refer the reader to [1] for more details.

B. Generalization to Complex-shaped Objects

In this work, we aim at generalizing **AGFP-Net** to objects with more complex shapes. One possibility could be to generate a new dataset with scenes containing complex-shaped objects, and retrain the neural network using the newly generated dataset. However, since the scene projections are constructed internally, building depth images of mesh objects can be time consuming, which defeats the purpose of accelerating TAMP. Also, this method does not guarantee the generalizability to unseen shapes during training. Therefore, we propose to represent complex-shaped objects at the feasibility prediction level using their bounding boxes. This allows the neural network to generalize to objects with complex shapes, without the need for any additional data generation/annotation and training effort to [1], and without an overhead in inference time.

Given an environment containing mesh objects, the tested action and the object to manipulate, we generate depth images as explained in Section IV-A and [1], using the bounding box of each object. These depth images are then given as input to **AGFP-Net** to obtain the probability of feasibility of performing the action on the object of interest, as well as the feasibility of each one of the defined grasp types. Since these grasp types represent continuous subsets of grasps, all grasps (obtained using an off-the-shelf grasp planner or from a grasp database) can be associated with one of these grasp types using simple geometric transforms (cf. Figure 2). The action a defined in (1) can be decomposed into $Pick$ and $Place$ sub-actions as follows:

$$a(r, O, s(O), \mathbf{q}) = Pick(r, O, s(O)) + Place(r, O, \mathbf{q}) \quad (2)$$

where $Pick$ corresponds to picking object O using r from $s(O)$, and $Place$ refers to placing O using robot r at $\mathbf{q}$. Following [1], we define the probability of feasibility p_F^a of the action as:

$$p_F^a = p_F^{Pick} \times p_F^{Place} \times max(\mathbf{p}_G^a) \quad (3)$$

where p_F^{Pick} and p_F^{Place} are the predicted probabilities of the $Pick$ and $Place$ sub-actions respectively, and $\mathbf{p}_G(a)$ is the vector of combined grasp type probabilities:

$$\mathbf{p}_G^a = \mathbf{p}_G^{Pick} \otimes \mathbf{p}_G^{Place} \quad (4)$$

$\otimes$ being the element-wise product.

These probabilities of feasibility of grasp types are used in (3) to ensure that there is at least one common grasp type between the $Pick$ and $Place$ actions. Additionally, during geometric planning, grasps are sorted according to the probability of feasibility of their corresponding grasp type. Motion planning is then performed on the most promising grasps first, allowing a faster planning time.

C. Generalization to Multi-Robot Settings

The design choices behind **AGFP-Net** [1] ensure that feasibility prediction is able to cover the whole workspace of a single robot. One important property of the model is that it is centered around the robot, meaning that it is dependent on the kinematics of the robot only, and does not depend on any other factors such as where the robot is placed in the environment. This is achieved by expressing all objects poses, fixed or movable, in the frame of the robot. Also, the scene projections are constructed such that the center of the robot's workspace is at the center of the depth images, and each projection covers the whole workspace of the robot. As a result, the scene projections only show the objects inside the robot's workspace.

These capabilities allow a smooth generalization of **AGFP-Net** to multi-robot problems. Indeed, the neural network can be queried for each robot individually by making sure the scene projections show the objects (or parts of

objects) in the workspace of the said robot only, and that the shown poses of these objects are in the frame of the latter. Given an object pose in the world frame $\mathbf{q}_O$ and a robot's frame transform $\mathbf{T}_r$, an object's pose expressed in the frame of the robot is simply $\mathbf{q}_O^r = \mathbf{T}_r \mathbf{q}_O$. Using this approach, **AGFP-Net** can be used to predict feasibility for each robot to move the objects inside its own workspace.

Moreover, this method allows testing composite actions such as $Pass$ actions. An object placed inside the intersection between two robots' workspaces would be visible in the scene projections corresponding to each robot, as shown in Figure 2. A $Pass$ action aims at placing an object at an intersection region for another robot to manipulate. Therefore, a $Pass$ action is feasible only if the first robot is able to pick the object from $s(O)$ and place it at the intersection placement q, and if the second robot is able to pick it from this placement. Given a $Pass$ action $a_{pass}(r_1, O, s(O), \mathbf{q})$ between two robots r_1 and r_2, we transform the pick and place poses to the frame of each robot to obtain $s(O)^{r_1}$, $\mathbf{q}^{r_1}$ and $\mathbf{q}^{r_2}$ [2]. Then, we compute the probability of feasibility $p_F^{a_{pass}}$ of the $Pass$ action between r_1 and r_2 as:

$$p_F^{a_{pass}} = p_F^a(r_1, O, s(O)^{r_1}, \mathbf{q}^{r_1}) \times p_F^{Pick}(r_2, O, \mathbf{q}^{r_2}) \quad (5)$$

For actions a_{goal} of type $goal$, and a_{temp} of type $Temp$ using a robot r:

$$p_F^{a_{goal}} = p_F^{a_{temp}} = p_F^a(r, O, s(O)^r, \mathbf{q}^r) \quad (6)$$

We develop a TAMP algorithm which uses these probabilities of feasibility to prioritize feasible actions and speedup the search of a geometrically feasible task plan.

V. Task and Motion Planning

We propose a feasibility-informed multi-robot TAMP algorithm capable of solving problems involving multiple robots. The main algorithm, shown in Algorithm 1, remains similar is based on a best-first tree search, where nodes represent states and edges are actions allowing transitions between states, following the definitions of Section III. A list Q of open nodes is maintained and sorted according to a defined cost. At each iteration, the node s with the minimum cost is extracted from Q and compared to the goal state s_{goal}. If s is a goal state, we retrieve the complete task plan leading to s then query the geometric planner to check the feasibility of every action in the task plan and plan their corresponding motions. If the action sequence is feasible, then a solution was found. Otherwise, the search continues. During geometric planning, the probabilities of feasibility of grasp types for each action are given to the geometric planning to prioritize feasible grasps.

If s in not a goal state, we expand the node using the function $findChildren$, shown in Algorithm 2. It first samples applicable actions at state s, by calling the function $findPossibleActions$ which is detailed in Algorithm 3. For each movable object O in the environment, we find the set of robots capable of reaching O at its pose in s, using the function $findReachingRobots(s(O), E)$. For each robot r_i in the obtained set, we generate $Goal$, $Pass$ and $Temp$ actions. In the case where O is not already at its goal pose, if s_{goal} is reachable by r_i, we sample a set of goal placements in the workspace of r_i. Otherwise, if the goal pose of O is not reachable by r_i, we try to generate $Pass$ placements.

Algorithm 1 Task and motion planner

Input: E, s_0, s_{goal} ▷ Environment, Initial and goal states
$Q \leftarrow \{s_0\}$ ▷ Set of nodes to expand
while Solution not found **do**
  $s \leftarrow argmin_{cost} Q$
  if $s \in s_{goal}$ **then**
    $[\tau, \Pi] \leftarrow retrieveSolution(s)$]
    if Π is feasible **then**
      return τ, Π
    end if
  else
    $Q \leftarrow Q \cup findChildren(s, E, s_{goal})$
  end if
end while

Algorithm 2 findChildren

Input: s, E, s_{goal}
$children \leftarrow \emptyset$
$A \leftarrow findPossibleActions(s, E, s_{goal})$
for each a in A **do**
  $[p_F^a, \mathbf{p}_G^a)] \leftarrow predictFeasibility(s, E, a)$
  $child \leftarrow nextState(s, a)$
  $child.cost \leftarrow computeCost(child, s_{goal}, p_F(a), E)$
  $children \leftarrow children \cup child$
end for
return $children$

We first find the set of robots that are able to reach s_{goal}. The planner needs to figure out how to bring O to each one of these robot's workspace. In order to do that, we build a graph $\mathbf{\Gamma}$ which vertices are all the robots in the environment, and edges represent the existence of an intersections between two robots' workspaces. For each robot r_g reaching s_{goal}, we use a Breadth-First Graph Search to find all possible paths from r_i to r_g, and we extract the first robot from each path. These are all the robots r_i can pass object O to in order to bring it closer to its goal pose. For each robot r_j in the obtained set of robots, we sample a number of $Pass$ placements at the intersection between $W(r_i)$ and $W(r_j)$.

Finally, we sample a set of $Temp$ placements in the workspace of r_i. Then, we generate actions corresponding to each one of the $Goal$, $Pass$ and $Temp$ placements sampled[3]. Once all possible actions are found, we call $predictFeasibility$ for each action a, which queries **AGFP-Net** and uses (3), (4) and (5) to compute the probability of feasibility of the action and the feasibility of the grasp types. We construct a new child as the result of applying a at s, we compute its cost and add it to the list of open nodes.

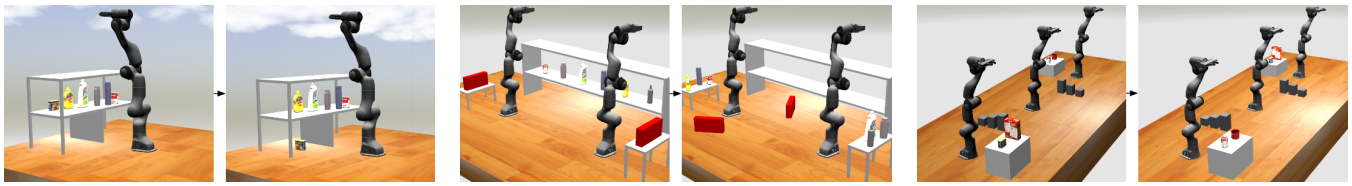
We define the cost of a node as:

$$C_{Total} = C_{SoFar} + C_{ToGoal} + C_{Feasibility} \quad (7)$$

where C_{SoFar} is the number of actions in the branch leading to the node, C_{ToGoal} is the minimum number of actions

[2]Note that $\mathbf{q}^{r_1}$ and $\mathbf{q}^{r_2}$ represent the placement pose expressed in the frame of robots r_1 and r_2 respectively, and are not robot configurations.

[3]Although the number of samples is a parameter fixed by the user, we give the planner the possibility to resample new placements, in case the previous ones do not lead to a feasible solution.



(a) *Access* problem (b) *Sort* problem (c) *Longswap* problem

Fig. 3: Visualization of the initial and goal states for 3 of the 6 problem domains used to test our approach.

Algorithm 3 findPossibleActions

Input: s, E, s_{goal}
$A \leftarrow \emptyset$
for each O in movable objects **do**
  for each r_i in $findReachingRobots(s(O), E)$ **do**
    $P \leftarrow \emptyset$
    if $s(O) \notin s_{goal}(O)$ **then**
      if $s_{goal}(O)$ is reachable by r_i **then**
        $P \leftarrow P \cup sampleGoal(r_i, E, s_{goal}(O))$
      else
        for r_g in $findReachingRobots(s_{goal}(O), E)$ **do**
          $next\_robots \leftarrow BFS(r_i, r_g)$
          for $r_j \in next\_robots$ **do**
            $P \leftarrow P \cup samplePass(r_i, r_j, E)$
          end for
        end for
      end if
    end if
    $P \leftarrow P \cup sampleTemp(r_i, E)$
    for each $\mathbf{q} \in P$ **do**
      $a \leftarrow Move(r_i, O, s(O) \rightarrow \mathbf{q})$
      $A \leftarrow A \cup a$
    end for
  end for
end for
return A

to reach the goal state, and $C_{Feasibility}$ is the feasibility cost detailed in [1]. In this work, C_{ToGoal} is constructed differently compared to [1] [33] in order to tackle multi-robot problems. Indeed, we take advantage of the previously constructed graph $\mathbf{\Gamma}$ to compute the shortest path length to the goal for each object using Breadth-First Graph Search. We then sum these path lengths to obtain the minimum number of actions to reach the goal state.

VI. Experiments

A. Test TAMP problems

In order to demonstrate the performance of our TAMP algorithm, we construct multiple TAMP problems with varying challenges in single, dual and triple robot settings. Each problem contains a number of complex-shaped objects extracted from the YCB [42] and KIT [43] objects databases. Note that none of these objects nor their bounding boxes were used during the training of **AGFP-Net**. Since grasp planning is outside the scope of this work, we manually annotate each object with up to 200 grasps.

We modify the *Access* and *Sort* domains defined [1] to measure the performance of our generalization to mesh objects and multi-robot settings. In the *Access* problem shown in Figure 3a, a single robot has to access and move a meat can. The difficulty is due to a number of objects blocking access to the wanted object, which requires removing all blocking objects to access the wanted one, before returning them to their initial pose. The *Sort* problem, illustrated in Figure 3b is a dual-robot problem where the goal is to sort objects on two pre-occupied tables. Here, the algorithm has to find feasible sets of placements on narrow surfaces. Moreover, each robot can reach one of the tables only.

We also define TAMP problems in which the presence of multiple robots might either make the problem easier or more challenging. The dual-robot *Assist* problem, shown in Figure 1b, is similar to the previously defined *Access* problem. For one robot, the red object is not accessible due to a number of blocking objects. The second robot, however, has direct access to it. This problem aims at testing the ability of our algorithm to find the simplest solution, which involves one robot assisting the other. In the *Clear* problem, illustrated in Figure 1a, two large objects (in red) span over the intersection region of two robots' workspaces, rendering *Pass* actions infeasible. The robots have to first clear the intersection region before moving a set of objects from one table to another.

Additionally, we test our approach on two three-robot problems, each with a different setting. The first is the *Middleman* problem (Figure 1c), in which three robots form a triangle such that there is an intersection between every pair of robots' workspaces. One of the intersection regions is blocked by a number of obstacles, forcing the robots to perform two *Pass* actions via a middleman robot to move two objects from one counter to the other. The second is the *Longswap* problem in which three sequential robots have to collaborate to swap the placements of 4 objects. For increased difficulty, the intersection regions between robots' workspaces are partially blocked by a set of obstacles.

B. Implementation details

We run our feasibility-informed TAMP algorithm on each one of these problems for 10 trials each, with a timeout set to 900 seconds per trial. We use Moveit! Task Constructor [44] for geometric planning with a BiTRRT motion planner [45]. On the feasibility prediction side, we reuse the neural network weights obtained using the training approach detailed in [1]. Experiments were conducted on an Intel i9-11950H @ 2.60GHz, with 32GB of RAM and NVIDIA RTX A3000 GPU. For comparison, we also run a baseline version our algorithm that does not leverage feasibility prediction, by setting all probabilities of feasibility to 1 during the search.

VII. Results

Table I shows the averaged results obtained for each problem. Results show that our feasibility-informed TAMP algorithm is able to solve all problems with 100% success rate, compared to the non-informed algorithm which fails

TABLE I: Planning performances with and without using feasibility prediction, averaged over 10 trials. Speedup is computed over all the trials by considering the timeout for failed cases and the average total planning time for successful ones.

Problem	Method	Success Rate (%)	Total Planning Time (s)	Geometric Planning Time (s)		Feasibility Prediction Time (s)	Infeasible Task Plans	Speedup
				Feasible actions	Infeasible actions			
Access	Baseline	0%	> 900	-	-	-	-	**> 11.7**
	Ours	**100%**	**76.8 (+/-38.3)**	**32.1 (+/-7.4)**	**2.5 (+/-1.8)**	**35.7 (+/-29.7)**	**2.9 (+/-1.5)**	
Sort	Baseline	0%	> 900	-	-	-	-	**> 11.1**
	Ours	**100%**	**81.0 (+/-25.5)**	**33.7 (+/-14.9)**	**18.6 (+/-15.3)**	**27.6 (+/-8.7)**	**1.9 (+/-2.2)**	
Assist	Baseline	90%	173.9 (+/-23.0)	116.6 (+/-15.8)	54.910 (+/-23.5)	-	66.4 (+/-4.0)	**> 35.7**
	Ours	**100%**	**6.9 (+/-5.6)**	**2.1 (+/-0.7)**	**1.6 (+/-3.4)**	**2.7 (+/-2.2)**	**0.7 (+/-1.2)**	
Clear	Baseline	90%	436.6 (207.4)	13.9 (+/-2.8)	421.7 (+/-207.8)	-	29 (9.3)	**> 14**
	Ours	**100%**	**34.5 (+/-24.3)**	**10.9 (+/-1.4)**	**13.5 (+/-21.7)**	**9.6 (+/-3.1)**	**0.6 (+/-0.9)**	
Middleman	Baseline	30%	520.7 (+/-49.5)	26.6 (+/-15)	493.9 (+/-35.8)	-	30.3 (+/-10.6)	**$> 55{,}7$**
	Ours	**100%**	**14.1 (+/-12.4)**	**5.9 (+/-1.9)**	**6.0 (+/-12.0)**	**2.0 (+/-0.5)**	**0.2 (+/-0.4)**	
Longswap	Baseline	20%	289.6 (+/-32.0)	31.7 (+/-2.0)	256.9 (+/-34.1)	-	14.0 (+/-1.0)	**> 6.5**
	Ours	100%	**118.7 (+/-41.1)**	**47.8 (+/-11.0)**	**7.0 (+/-13.3)**	**54.7 (+/-18.2)**	**1.1 (+/-1.1)**	

at least once, and completely fails to solve the single-robot $Access$, and the dual-robot $Sort$ problems. Given the high combinatorial complexity of these problems, this outcome can be expected and shows that our proposed method is able to efficiently filter the tree search and reach a solution faster, allowing a total planning time of $76.8s$ for the $Access$ problem and $81s$ for the $Sort$ problem. These results are also comparable to the ones obtained in [1], which shows that our method is able to handle complex-shaped objects and multi-robot systems without hurting the planning time.

For the $Assist$ problem, using **AGFP-Net** allows at least a 35 times speedup in planning time. In this problem, the non-informed planner is sensitive to the order in which objects are processed during the search. If the action moving the red object to a $Pass$ region (using the robot with easy access to it) appears early in the search, planning time can be low. Our approach removes this sensitivity to the order of objects by prioritizing actions according to their feasibility. Results on the $Clear$ problem, on the other hand, show a 92% reduction in planning time using feasibility prediction. This gain in performance is obtained thanks to our proposed method for evaluating the probability of $Pass$ actions. Indeed, since the intersection region between two robots' workspaces is not fully covered by the red objects, both the $Pick$ and the $Place$ actions of the passing robot might be feasible. However, taking into account the feasibility of the following $Pick$ action using the receiving robot allows the planner to prioritize clearing the intersection region before performing $Pass$ actions between the robots. This is demonstrated by the reduction in infeasible task plans generated. The three-robot $Middleman$ problem presents a similar scenario, except the objects blocking the intersection region are fixed obstacles. Also, the added robot adds to the combinatorial complexity of the problem. Results show an improvement in success rate from 30% without feasibility prediction to 100% using **AGFP-Net**. Also, total planning time using the latter is at least 55 times faster than when no heuristic is used. This shows that our feasibility-informed planner avoids spending extensive effort on trying to perform a single $Pass$ action directly to the goal robot, and prefers the use of a middleman robot with two $Pass$ actions for each object.

The performance yielded on the $Assist$, $Clear$ and $Middleman$ problems show that using feasibility prediction, our TAMP algorithm is able to identify scenarios where robot collaboration is advantageous or necessary, generally resulting in the first generated task plan being geometrically feasible as shown in Table I. Results on the $Longswap$ problem demonstrate the ability of our approach to generalize to different multi-robot settings. In addition to an improvement in success rate from 20% to 100%, our feasibility-informed planner reduces the total planning time by at least 84%. Using **AGFP-Net**, our algorithm identifies which of the sampled $Pass$ placements are free and prioritizes them. It also recognizes occupied goal placements and includes $Temp$ placements in its solution in order to free them.

These results show that our proposed approach is able to tackle single and multi-robot TAMP problems involving complex shaped objects. Feasibility prediction not only guarantees 100% success rate on all problems, but it also reduces considerably the planning time. Also, the overhead due to feasibility prediction is largely compensated by the time saved in geometric planning time.

VIII. Conclusion

In this paper, we present a method for extending the use of **AGFP-Net** [1] to complex-shaped objects, allowing action and grasp feasibility prediction on realistic objects. We also propose a framework for predicting the feasibility of actions in arbitrary multi-robot settings, taking advantage of the robot-centric nature of the neural network, and using a new approach for computing the probability of feasibility of collaborative actions such as $Pass$ actions. Moreover, we develop a feasibility-informed multi-robot TAMP algorithm, capable of solving complex TAMP problems involving multiple robots. We demonstrate the performance of our method on six TAMP problems containing multiple complex-shaped objects, and different single and multi-robot settings. Results show a notable gain in success rate and planning time using feasibility prediction as a heuristic. Future work might involve extending the approach to coordinated motion planning problems, and handling more collaborative actions such as handovers, allowing a parallel execution of tasks.

REFERENCES

[1] S. A. Bouhsain, R. Alami, and T. Simeon, "Simultaneous action and grasp feasibility prediction for task and motion planning through multi-task learning," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023.

[2] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual review of control, robotics, and autonomous systems*, vol. 4, pp. 265–293, 2021.

[3] A. M. Wells, N. T. Dantam, A. Shrivastava, and L. E. Kavraki, "Learning feasibility for task and motion planning in tabletop environments," *IEEE robotics and automation letters*, vol. 4, no. 2, pp. 1255–1262, 2019.

[4] D. Driess, O. Oguz, J.-S. Ha, and M. Toussaint, "Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9563–9569.

[5] L. Xu, T. Ren, G. Chalvatzaki, and J. Peters, "Accelerating integrated task and motion planning with neural feasibility checking," *arXiv preprint arXiv:2203.10568*, 2022.

[6] Z. Yang, C. R. Garrett, and D. Fox, "Sequence-based plan feasibility prediction for efficient task and motion planning," *arXiv preprint arXiv:2211.01576*, 2022.

[7] R. Alami, T. Simeon, and J.-P. Laumond, "A geometrical approach to planning manipulation tasks. the case of discrete placements and grasps," in *The fifth international symposium on Robotics research*. MIT Press, 1990, pp. 453–463.

[8] Y. Koga and J.-C. Latombe, "On multi-arm manipulation planning," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE, 1994, pp. 945–952.

[9] J. M. Ahuactzin, K. Gupta, and E. Mazer, "Manipulation planning for redundant robots: a practical approach," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 731–747, 1998.

[10] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 729–746, 2004.

[11] K. Hauser and J.-C. Latombe, "Multi-modal motion planning in non-expansive spaces," *The International Journal of Robotics Research*, vol. 29, no. 7, pp. 897–915, 2010.

[12] K. Hauser, "Randomized belief-space replanning in partially-observable continuous spaces," in *Algorithmic Foundations of Robotics IX*. Springer, 2010, pp. 193–209.

[13] K. Hauser and V. Ng-Thow-Hing, "Randomized multi-modal motion planning for a humanoid robot manipulation task," *The International Journal of Robotics Research*, vol. 30, no. 6, pp. 678–698, 2011.

[14] J. Barry, K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Manipulation with multiple action types," in *Experimental Robotics*. Springer, 2013, pp. 531–545.

[15] J. Barry, L. P. Kaelbling, and T. Lozano-Pérez, "A hierarchical approach to manipulation with diverse actions," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1799–1806.

[16] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," *The International Journal of Robotics Research*, vol. 28, no. 1, pp. 104–126, 2009.

[17] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 639–646.

[18] F. Lagriffoul, D. Dimitrov, J. Bidot, A. Saffiotti, and L. Karlsson, "Efficiently combining task and motion planning using geometric constraints," *The International Journal of Robotics Research*, vol. 33, no. 14, pp. 1726–1747, 2014.

[19] L. De Silva, M. Gharbi, A. K. Pandey, and R. Alami, "A new approach to combined symbolic-geometric backtracking in the context of human-robot interaction," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 3757–3763.

[20] M. Gharbi, R. Lallement, and R. Alami, "Combining symbolic and geometric planning to synthesize human-aware plans: toward more efficient combined search," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6360–6365.

[21] J. Bidot, L. Karlsson, F. Lagriffoul, and A. Saffiotti, "Geometric backtracking for combined task and motion planning in robotic systems," *Artificial Intelligence*, vol. 247, pp. 229–265, 2017.

[22] C. R. Garrett, T. Lozano-Perez, and L. P. Kaelbling, "Ffrob: Leveraging symbolic planning for efficient task and motion planning," *The International Journal of Robotics Research*, vol. 37, no. 1, pp. 104–136, 2018.

[23] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Sampling-based methods for factored task and motion planning," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1796–1825, 2018.

[24] R. Chitnis, D. Hadfield-Menell, A. Gupta, S. Srivastava, E. Groshev, C. Lin, and P. Abbeel, "Guided search for task and motion plans using learned heuristics," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 447–454.

[25] J. Carpentier, R. Budhiraja, and N. Mansard, "Learning feasibility constraints for multi-contact locomotion of legged robots," in *Robotics: Science and Systems*, 2017, p. 9p.

[26] 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.

[27] B. Kim and L. Shimanuki, "Learning value functions with relational state representations for guiding task-and-motion planning," in *Conference on Robot Learning*. PMLR, 2020, pp. 955–968.

[28] D. Driess, J.-S. Ha, and M. Toussaint, "Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image," *arXiv preprint arXiv:2006.05398*, 2020.

[29] D. Xu, A. Mandlekar, R. Martín-Martín, Y. Zhu, S. Savarese, and L. Fei-Fei, "Deep affordance foresight: Planning through what can be done in the future," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6206–6213.

[30] T. Silver, R. Chitnis, A. Curtis, J. B. Tenenbaum, T. Lozano-Pérez, and L. P. Kaelbling, "Planning with learned object importance in large problem instances using graph neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 13, 2021, pp. 11 962–11 971.

[31] B. Kim, L. Shimanuki, L. P. Kaelbling, and T. Lozano-Pérez, "Representation, learning, and planning algorithms for geometric task and motion planning," *The International Journal of Robotics Research*, vol. 41, no. 2, pp. 210–231, 2022.

[32] M. J. McDonald and D. Hadfield-Menell, "Guided imitation of task and motion planning," in *Conference on Robot Learning*. PMLR, 2022, pp. 630–640.

[33] S. Ait Bouhsain, R. Alami, and T. Simeon, "Learning to predict action feasibility for task and motion planning in 3d environments," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[34] M. Khodeir, B. Agro, and F. Shkurti, "Learning to search in task and motion planning with streams," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 1983–1990, 2023.

[35] S. Park, H. C. Kim, J. Baek, and J. Park, "Scalable learned geometric feasibility for cooperative grasp and motion planning," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 545–11 552, 2022.

[36] L. Antonyshyn, J. Silveira, S. Givigi, and J. Marshall, "Multiple mobile robot task and motion planning: A survey," *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–35, 2023.

[37] I. Umay, B. Fidan, and W. Melek, "An integrated task and motion planning technique for multi-robot-systems," in *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*. IEEE, 2019, pp. 1–7.

[38] H. Karami, A. Thomas, and F. Mastrogiovanni, "Task allocation for multi-robot task and motion planning: A case for object picking in cluttered workspaces," in *International Conference of the Italian Association for Artificial Intelligence*. Springer, 2021, pp. 3–17.

[39] V. N. Hartmann, A. Orthey, D. Driess, O. S. Oguz, and M. Toussaint, "Long-horizon multi-robot rearrangement planning for construction assembly," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 239–252, 2022.

[40] H. Zhang, S.-H. Chan, J. Zhong, J. Li, S. Koenig, and S. Nikolaidis, "A mip-based approach for multi-robot geometric task-and-motion planning," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2022, pp. 2102–2109.

[41] J. Motes, T. Chen, T. Bretl, M. M. Aguirre, and N. M. Amato, "Hypergraph-based multi-robot task and motion planning," *IEEE Transactions on Robotics*, 2023.

[42] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks

for manipulation research," in *2015 international conference on advanced robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[43] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.

[44] M. Görner, R. Haschke, H. Ritter, and J. Zhang, "Moveit! task constructor for task-level motion planning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 190–196.

[45] D. Devaurs, T. Siméon, and J. Cortés, "Enhancing the transition-based rrt to deal with complex cost spaces," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4120–4125.