Task-Centric Selection of Robot and Environment Initial Configurations for Assistive Tasks

Ariel Kapusta*, Daehyung Park, and Charles C. Kemp

***Abstract*— When a mobile manipulator functions as an assistive device, the robot's initial configuration and the configuration of the environment can impact the robot's ability to provide effective assistance. Selecting initial configurations for assistive tasks can be challenging due to the high number of degrees of freedom of the robot, the environment, and the person, as well as the complexity of the task. In addition, rapid selection of initial conditions can be important, so that the system will be responsive to the user and will not require the user to wait a long time while the robot makes a decision. To address these challenges, we present Task-centric initial Configuration Selection (TCS), which unlike previous work uses a measure of task-centric manipulability to accommodate state estimation error, considers various environmental degrees of freedom, and can find a set of configurations from which a robot can perform a task. TCS performs substantial offline computation, so that it can rapidly provide solutions at run time. At run time, the system performs an optimization over candidate initial configurations using a utility function that can include factors such as movement costs for the robot's mobile base. To evaluate TCS, we created models of 11 activities of daily living (ADLs) and evaluated TCS's performance with these 11 assistive tasks in a computer simulation of a PR2, a robotic bed, and a model of a human body. TCS performed as well or better than a baseline algorithm in all of our tests against state estimation error.**

I. Introduction

When faced with the problem of having a mobile robot perform a manipulation task, the first step is often to select a configuration or configurations from which to perform the task. With a good initial configuration, the robot is more likely to be able to complete the task successfully. Selecting a good initial configuration can be challenging and has been addressed in many ways. In our previous work, we have selected start locations for the robot manually with trial and error [1], [2], or have used data-driven approaches using robot-centric success likelihood [3]. In this work we take a different approach: we present Task-centric initial Configuration Selection (TCS) (see Figure 1). With a task-centered focus, we can use specifics of the problem to aid the robot in finding solutions. Tasks can be challenging if there are many degrees of freedom in an environment to customize, or if they require multiple initial configurations to complete. Our task-centric approach helps address these issues.

Toward a mobile manipulator assisting motor-impaired users with activities of daily living (ADLs), we applied TCS to representations of such tasks. We created models (see section III-A.2) for 11 ADL tasks on which to evaluate our method. We found that TCS returns a solution set of initial configurations from which the robot can perform the task despite error in state estimation. TCS uses task-centric reachability and task-centric manipulability scores to determine its solution. For simplicity, we will refer to our *task-centric manipulability score* as *TC-manipulability* in the remainder of this paper. Similiarly, we will refer to our *task-centric reachability score* as *TC-reachability*.

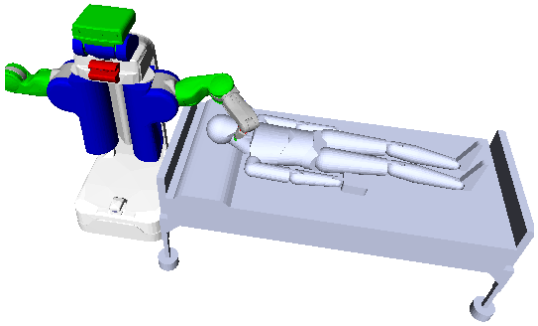


Fig. 1: Simulation environment of a PR2 robot and a human in a configurable bed, showing an initial configuration to shave the user in the bed. TCS finds that putting the robot behind the bed can be effective.

Our method differs from previous research by being task-centric rather than robot- or object- centric. TCS has three primary features that distinguish it from related works: the TC-manipulability measure, consideration of sets of multiple configurations as a solution, and generalization to environmental degrees of freedom.

As in most real-world robotics applications, when performing assistive tasks with a robot, the robot estimates the world's relevant current state. For assistive tasks, the relevant states may be the pose of the user's head, the robot's current position, the location of the wheelchair, etc., and there may be estimation errors. Our approach to selecting initial configurations depends on rough modeling and heavy precomputation, so our method must be robust to state estimation error. We use TC-reachability and TC-manipulability scores based on kinematic isotropy (from [4]) to make our method robustness to such errors.

We found from our previous work in [1] that some tasks require that the robot use multiple base positions. When shaving, for example, we found that the robot could not shave the entirety of the face (both sides) from only one position. Note that in this paper, the term configuration refers to both the robot's position and to the configuration of any configurable environmental objects. Our method considers

A. Kapusta, D. Park, and C. C. Kemp are with the Healthcare Robotics Lab, Georgia Institute of Technology, Atlanta, GA. *A. Kapusta is the corresponding author (akapusta@gatech.edu).

the opportunity of using multiple configurations to perform a single task.

If the environment is configurable in a way that can impact performance of a task, the robot may improve its performance by adjusting the environment appropriately. For example, if a user with motor impairments is on a configurable bed, the robot might be better able to shave the user in a seated position rather than a supine position. Our method can be expanded to allow additional configurable degrees of freedom. In section IV-A.2 we describe our evaluation of TCS on an environment with 8 configurable degrees of freedom, including those for the robot and the bed. Our results suggest that having a user in a configurable bed, preferably a robotic bed that can be controlled along with the mobile manipulator, could facilitate the performance of assistive tasks. As can be seen in section V or specifically in figure 6, our TCS algorithm uses the capabilities of the configurable bed to perform the tasks.

II. Related Work

Several bodies of work have examined the proxemics of human-robot interactions [5], [6], [7], [8], [9]. These works look at acceptable interpersonal distances between humans and robots in social settings. For this paper, we do not consider proxemics or social factors and instead focus on physical aspects of the task.

Various works have used the concepts of human-robot proxemics to inform the robot when performing tasks. These works couple task performance concepts with scoring methods based on proxemics to select base positions and paths for the robot and item handover locations [10], [11], [12], [13]. A thorough survey of human-aware robot navigation can be found in [14]. We do not address navigation paths and concentrate on finding fixed final goal configurations for the mobile robot's and environment to perform tasks. We also focus specifically on assistive tasks.

There has been previous work in creating models of human activities to inform assistive robots. Redmond et al. collected force and torque data as users performed various activities of daily living and characterized the tasks by their haptic characteristics [15]. Hawkins et al. used contact forces from participants wiping and shaving to create task models to inform a robot of acceptable and anomalous force characteristics when performing the task [1]. Jain and Kemp present a method for improving robot manipulation by creating data-driven object-centric force models of tasks [16]. For this paper, we use a simple kinematic task model that represents a task as a sparse set of goal poses for the robot's end effector.

Prior research has investigated how to find good poses for a robot's mobile base. Hsu et al. presented a method for selecting a place for an industrial robot manipulator to perform a series of tasks amidst clutter. They used randomized path planners to generate collision free paths for the arm and they randomly perturb the robot position to find positions from which the tasks can be performed with low performance time. However, their approach takes minutes to select a robot position and is not intended for real-time use

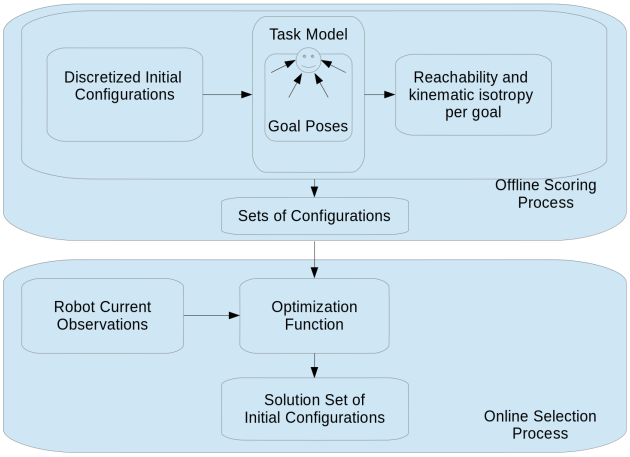


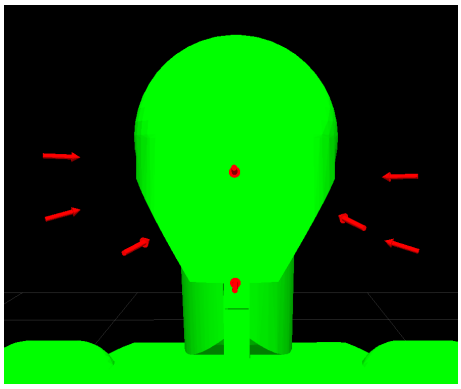
Fig. 2: The workflow used in TCS.

[17]. Zacharias et al. 2007 introduced and used a method for representing and scoring the workspace of a robot to create a capability map. The capability map is the discretized 3-D workspace around a robot arm, scored by the number of discretized orientations the end effector can reach at each discretized 3-D point [18]. Various works have used this robot-centric capability map to select the robot base position by overlapping the capability map with end-effector goal poses [19], [20], [21]. Leidner et al. also addresses regions of interest and integration with a whole-body controller (doing whole-body inverse-kinematics) once the task is in progress [21]. Vahrenkamp et al. inverts the capability map from [18] to create a grasp-centric scoring method. For a given grasp, they assign a score to potential robot base positions around the grasp; the score is the value for that end-effector goal pose from the capability map. They overlay a scoring map for each goal grasp to select the robot base position with the highest score [22].

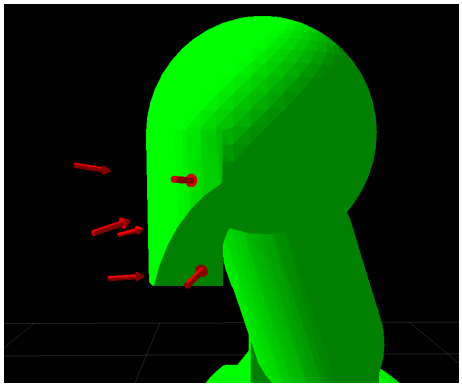
In contrast, we use two different scoring methods, one based on finding an inverse-kinematics solution, and one based on kinematic isotropy from [4]. Our configuration selection is task-centric, generating a TC-manipulability and TC-reachability score for each set of initial configurations in a the task-centric map. By making our method task-centric, the system can focus on the specifics of the problem, such as including the additional degrees of freedom when assisting a user in a configurable bed. Our method discretizes the initial configuration space instead of the arm's workspace, and it keeps goal poses in continuous space. Our method also returns the solution set of initial configurations rapidly with its real-time module.

Stulp et al. present what can be described as a task-centric method for selecting the areas in which to place a mobile manipulator from which it can perform a grasping task. Their method uses Monte-Carlo simulation with introduced pose uncertainty to find base positions with high success rates. In essence, they simulate the robot performing a cup grasping task from various base positions and score the base positions on their success rates. For real-time base position selection, they convolve uncertainty in robot location with base position scores to provide an area of high-success

probability. However, their method is used with only 2 degrees of freedom (x and y of base position) and for a single task consisting of a single grasp pose on a cup in a varying location [23]. In contrast, we use robot kinematics to score initial configurations, instead of running simulations. We consider many degrees of freedom in the initial configuration space and we consider sets of initial configurations.



(a)



(b)

Fig. 3: The goal poses for the shaving task displayed on the wheelchair model. Each arrow represents a position and orientation, 6-DoF end-effector goal pose. (a) viewed from the front (b) viewed from the side.

III. Method

We will first explain the framework of TCS. Afterwards we will explain some of the specifics of our implementation.

A. Framework

1) A Good Configuration:

An important question to ask is: "What makes an initial configuration *good*?"

In this paper we use the term *initial configuration* to mean the position and orientation of the robot's mobile base, the z-axis spine height of the robot, and any additional environmental degrees of freedom that are adjustable. We also consider initial configurations in sets that can be of cardinality 1 or greater. The goal of selecting initial configurations is to allow the robot to perform a task. The robot should be able to perform the task from a *good* set of initial configurations. We judge the robot's ability to perform the task from a set of initial configurations with one measure: the percent of goal poses its end effector can achieve without the robot being in collision. If the robot can find an inverse kinematics (IK) solution to all the goal poses associated with a task, we judge that the robot can perform the task. However, there is modeling and state estimation error that may foil the robot, and the robot does not know how the error will manifest apriori. From a good set of initial configurations, the robot should be able to perform the task despite such error. Our TCS algorithm uses two scores, TC-reachability and TC-manipulability, and an optimization over a utility function to select a set of configurations for a task.

2) Task Modeling:

Our aim with task modeling is to create a representation that allows a robot to efficiently make decisions about its ability to perform a task. We model each task as a sparse set of positions and orientations (Cartesian position and quaternion) for the robot's end effector. With this representation, we assume that if the robot can reach all the desired end-effector poses, it will be able to perform the task. We limited tasks to one-handed tasks and used only the robot's left arm in our evaluation.

3) Nomenclature:

c: A task identifier

N_c: Number of goal poses for task class c

x_i: A position and orientation end-effector goal pose. $x_i \in \mathbb{R}^6$

$\boldsymbol{x}_c$: Set of goal poses, $\{x_1, x_2, ..., x_{N_c}\}$ for task class c.

q_j: A joint configuration of the robot arm. $q_j \in \mathbb{R}^n$, where n is the number of DoF of the arm

h_i: An initial configuration of the robot and environment

h_0: The current configuration of the robot and environment

$\boldsymbol{q}_{x_i,h_j}$: Set of IK joint configuration solutions to goal x_i from initial configuration h_j, $\{q_1, q_2, ..., q_n\}$, where n is the number of IK solutions

$\boldsymbol{h}_k$: A set of of cardinality $>= 1$ of initial configurations of the robot base and configurable environment, $\{h_1, h_2, ..., h_n\}$, where n is the number of initial configurations in set $\boldsymbol{h}_k$

τ: The set of initial configuration sets, $\{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_n\}$, where n is the number of sets of initial configurations being considered.

η: The set of valid discretized configurations

$J(q)$ The Jacobian of the arm in joint configuration q

a: The order of the robot arm. In our case, 6.

$\Delta(q_j)$: The kinematic isotropy for the arm in joint configuration q_j

4) Precomputed Scoring: TC-Reachability and TC-Manipulability:

At the core of TCS, we use two scores, TC-reachability and TC-manipulability, to determine if the robot is likely to be able to perform an assistive task using a set of initial configurations. This scoring process is computationally expensive and is computed offline. The saved scores can then be referenced quickly online. We limit interaction to

the PR2's left arm. The algorithm could be extended in a straightforward manner to use either or both arms.

We first discretize the initial configuration space into η. If we would like to impose constraints on the initial configurations (e.g. require that the user be seated upright in the configurable bed), we apply these constraints to η.

The discretization size and resolution directly affects the computational cost of the scoring and the memory cost of the saved scores. Section IV describes the discretization we used in our evaluation and how we limited the practical computational cost.

We create τ by selecting all sets of one or more initial configurations from η and we evaluate a *TC-reachability* and *TC-manipulability* score for each $\boldsymbol{h}_k$ in τ.

Our terms *TC-reachability* and *TC-manipulability* differ from common terms found in other works, and we define them below. See section III-A.3 for clarification on nomenclature.

Our TC-reachability score P_R for a set of initial configurations $\boldsymbol{h}_k$ and a task c is shown in equation (1).

$$P_R(\boldsymbol{h}_k, c) \quad = \quad (\frac{1}{N_c}) \sum_{i=1}^{N_c} \max_{h_j \in \boldsymbol{h}_k} W(h_j, i), \qquad (1)$$

where $W(h_j, i)$ is a binary function. $W(h_j, i) = 1$ if the robot finds an IK solution to the goal pose j from initial configuration h_i; otherwise $W(h_j, i) = 0$.

Our TC-manipulability score uses kinematic isotropy from [4], which we define now. The equation for kinematic isotropy is shown in equation (2). $\Delta(q_i)$ is the kinematic isotropy for the arm in joint configuration q_i.

$$\Delta(q_i) \quad = \quad \frac{\sqrt[a]{\det(J(q_i)J(q_i)^T)}}{(\frac{\text{trace}(J(q_i)J(q_i)^T)}{a})}, \qquad (2)$$

where $J(q_i)$ is the Jacobian of the robot arm in joint configuration q_i and a is the order of the robot arm (6 in the case of our 7-DoF arm).

We use a function, $F(x_i, h_j)$ to find the maximum value of $\Delta(q_i)$ for goal pose x_i.

$$\begin{aligned} F(x_i, h_j) \quad &= \quad \max_{q_j \in \boldsymbol{q}_{x_i,h_j}} (\Delta(q_j)) \quad \forall \boldsymbol{q}_{x_i,h_j} \notin \emptyset; \\ \text{or} \qquad & \qquad (3) \\ F(x_i) \qquad &= 0 \qquad\qquad\qquad\quad \forall \boldsymbol{q}_{x_i,h_j} \equiv \emptyset \end{aligned}$$

The calculation for our TC-manipulability score $P_M(\boldsymbol{h}_k, c)$ is shown in equation (4).

$$P_M(\boldsymbol{h}_k, c) \quad = \quad (\frac{1}{N_c}) \sum_{i=1}^{N_c} \max_{h_j \in \boldsymbol{h}_k} F(x_i, h_j). \qquad (4)$$

Our TC-reachability and TC-manipulability scores each range from 0 to 1.

5) Online Selection:

When running our real-time service, we want a rapid response to the user from the algorithm with a good set of initial configurations for the task, given the current situation. TCS loads the precomputed scores and runs a brute force optimization on equation (5).

$$\arg\max_{\boldsymbol{h}_k} \quad \alpha P_R(\boldsymbol{h}_k, c) + \beta P_M(\boldsymbol{h}_k, c) - \gamma D(\boldsymbol{h}_k, h_0), \quad (5)$$

where α, β, and γ are gains in the optimization function, adjustable depending on the priorities of the user. $D(\boldsymbol{h}_k, h_0)$ is a cost function based on the robot's current location h_0 and the set of initial configurations h_k. For example, this can be based on Euclidean distance. Although, we did not formally evaluate it, we have found that with $D(\boldsymbol{h}_k, h_0)$ as a Euclidean distance cost on the robot's X and Y position, this step completes rapidly, on the order of 1 second.

B. The Process

Figure 2 shows the workflow of TCS.

1) Discretize the initial configuration space into η.
2) Model tasks as goal end-effector poses in reference to relevant coordinate frames.
3) For each configuration, evaluate which goal poses are reachable and calculate the kinematic isotropy $\Delta(q_i)$.
4) Create the sets of initial configurations $\boldsymbol{h}_k$ which make up τ, and evaluate their respective TC-reachability (P_R) and TC-manipulability (P_M) scores.
5) Save scores
6) ROS service call requests a set of initial configurations for a task.
7) Consider current robot observations.
8) Using optimization function, select and return the best set of initial configurations

TABLE I: Results of running TCS in 1000 Monte-Carlo simulations. Mean and standard deviation (std) of percent reached goals with error introduced into the location of the human and the wheelchair.

Task	Model	Performance: Mean (std)
Shaving	Wheelchair	99.9% (1.6)
Feeding	Wheelchair	95.5% (14.4)
Brushing	Wheelchair	100.0% (0.0)
Shaving	Configurable Bed without Wall	97.0% (8.9)
Shaving	Configurable Bed	99.9% (1.4)
Bathing	Configurable Bed	79.5% (6.5)
Scratching Upper Arm Left	Configurable Bed	66.0% (39.8)
Scratching Upper Arm Right	Configurable Bed	100.0% (1.6)
Scratching Forearm Left	Configurable Bed	100.0% (0.0)
Scratching Forearm Right	Configurable Bed	100.0% (0.0)
Scratching Thigh Left	Configurable Bed	99.8% (2.7)
Scratching Thigh Right	Configurable Bed	99.7% (3.9)
Scratching Chest	Configurable Bed	74.4% (37.4)

IV. Evaluation

A. Implementation

We manually selected the goal poses for each task, which we defined with respect to relevant reference frames (e.g. the

head for shaving, or the shoulder for scratching the upper arm). A goal pose is a position and orientation goal for the robot's end effector. As described in section III-A.2 each task was represented with a model consisting of a set of goal poses. We created task models for various activities of daily living for which a robot like the PR2 may be able to provide assistance. The tasks we modeled were: shaving, feeding, brushing hair, bathing (sponge bath), and scratching the left/right upper arm, left/right forearm, left/right thigh, and chest (each scratching task was considered separately). For example, figure 3 shows the goal poses for the shaving task.

We used OpenRave (http://www.openrave.org/) to simulate environments in which to test TCS. We created an environment in OpenRave with a PR2 robot and a model of an average male human which we placed either in a wheelchair or in a configurable bed. The human model dimensions come from [24]. The PR2 is a mobile manipulator made by Willow Garage with two 7-DoF arms. We created the models for the configurable bed and the wheelchair to match the Invacare 5401IVC full electric hospital bed and a wheelchair we have in our lab, respectively. We used OpenRave's inverse kinematics database for the PR2 robot to find IK solutions to use in our algorithm. When exploring multiple initial configurations, we assume the robot can move from one configuration to another.

To save on computation time, memory, and file size, we only considered sets of configurations for which each configuration could reach at least one goal pose, and for which the configurations were a minimum distance apart. We applied this minimum distance only to the X-Y position of the base. We also only saved sets of configurations if their TC-manipulability score was greater than the highest TC-manipulability of configuration sets with lower cardinality and we limited the maximum cardinality of initial configuration sets to 2. We used joblib (https://pythonhosted.org/joblib/) to assist in saving and loading files.

We used the Robot Operating System (ROS) to handle communication between a user and the robot/algorithm. We used the ROS service protocol to handle the user's request. The service receives as input a semantic description of the task and looks for task relevant coordinate frames in ROS's transform (tf) tree. For the shaving task, for example, it only looks for a head pose, but for bathing, it uses a coordinate frame for each limb. Getting such frames for a real user in bed may be difficult; automated body pose estimation could be useful. When evaluating the system in simulation, the simulator provides these coordinate frames.

1) Evaluation in Wheelchair Environment:
Figure 4 shows the simulation environment with the PR2 and human in wheelchair.

We used TCS to determine the TC-manipulability and TC-reachability of initial configurations for several tasks (brushing, feeding, shaving were done using the wheelchair model).

We allowed TCS to explore four configurable degrees of freedom: robot X and Y position on the ground plane, robot base orientation on the ground plane, and Z-axis height of the robot spine. We discretized these degrees of freedom into 5 cm, 5 cm, 45°, and 15 cm increments, respectively. The DoF ranges were [-1.5m, 1.5m], [-1.5m, 1.5m], [0, 360°), and [0, 30cm] respectively.

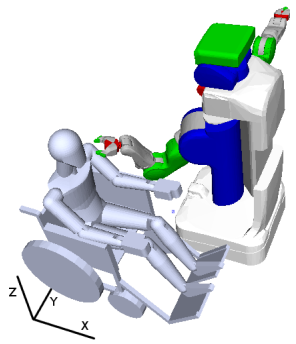


Fig. 4: The OpenRave environment with PR2 and wheelchair. A simulation screenshot from the shaving task, with global axes manually added.

A TC-reachability score of 1 implies that the robot can reach all goal poses from every set of initial configurations . We evaluated each set of initial configurations by introducing normally distributed error into the location of the human and wheelchair and seeing what percent of the goal poses for the task the robot could still find a valid IK solutions. We ignore collisions between the robot base and the wheelchair for this evaluation. We ran this as a Monte Carlo simulation 1000 times. In each simulation, we selected the best set of initial configurations for the task, then introduced error by sampling the normally distributed error. The error distributions are shown in table II. Error was introduced by moving the wheelchair in the X and Y directions and rotating it about the Z axis. We also moved the human with respect to the wheelchair in the X and Y directions, rotated it about the Z axis, and tilted the head up and down.

From the 1000 simulations and the percent of goal poses the robot could reach in each, we created statistics on the effectiveness of the initial configuration for the task.

2) Evaluation in Configurable Bed Environment:
Figure 5 shows the simulation environment with a PR2 and with a human on a configurable bed. We put a wall behind the bed to emulate how beds are often positioned in rooms. We also tested an environment without a wall behind the bed

TABLE II: Normally distributed pose estimation error imposed on Monte-Carlo simulation of the task.

		Error
Environment	Error DoF	Mean (std)
Wheelchair	Wheelchair: global X direction	0 (1.25cm)
	Wheelchair: global Y direction	0 (1.25cm)
	Wheelchair: global Z Rotation	0 (2.5deg)
	Human: local X direction	0 (5cm)
	Human: local Y direction	0 (5cm)
	Human: global Z Rotation	0 (5deg)
	Human: Head Tilt	0 (5deg)
Configurable Bed	Human: global X direction	0 (5cm)
	Human: global Y direction	0 (5cm)

with the shaving task.

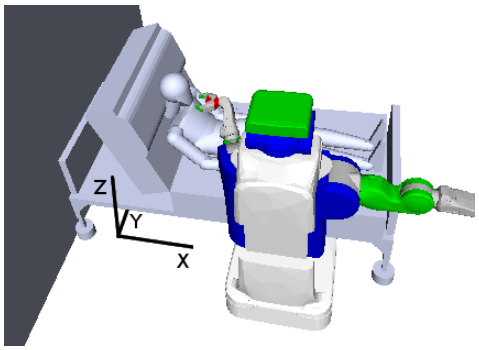


Fig. 5: The OpenRave environment with PR2 and configurable bed: a simulation screenshot from the shaving task, with global axes manually added.

We then used TCS to determine the TC-manipulability and TC-reachability of possible initial configurations for the following tasks: shaving, bathing, and scratching left upper arm, scratching right upper arm, scratching left forearm, scratching right forearm, scratching left thigh, scratching right thigh, and scratching chest.

We allowed TCS to explore six configurable degrees of freedom: robot X and Y position on the ground plane, robot base orientation on the ground plane, Z-axis height of the robot spine, bed Z-axis height, and tilt of the head rest. We discretized these degrees of freedom into 10 cm, 10 cm, 90°, 15 cm, 10 cm, and 35° increments, respectively. The DoF ranges were [-2.0m, 2.0m], [-2.0m, 2.0m], [0, 360°), [0cm, 30cm], [0cm, 30cm], and [0, 70°), respectively. TCS also looked at shifts of the human in the Y direction (discretized into 5 cm increments, ranging [-10cm, 10cm]), but used the human's Y position to essentially define separate tasks. When selecting a best set of initial conditions, TCS would use the task with the human's position best represented.

As for the wheelchair environment, we ran this as a Monte Carlo simulation 1000 times. In each simulation we selected the best set of initial configurations for the task, then introduced error by selecting a sample from the normally distributed error. The error distributions are shown in table II. Error was introduced by moving the human in the X and Y directions with respect to the bed.

V. Results and Discussion

The results of our tests showed that the initial configurations suggested by TCS can work well for performing tasks despite introduced state estimation error. See Table I for tabulated results. There are several limitations to using this measure, which are described in section V-A.

Table III shows a 2-way comparison of the effect of the use of TC-manipulability and multiple initial configurations on the task performance for several tasks. For this analysis, when not using TC-manipulability, there are many sets of configurations with the same TC-reachability score, too many to evaluate. We assumed a uniform distribution from the sets of initial configurations with equal TC-reachability from which we could sample. We performed 1000 Monte-Carlo simulations for each test. The performance is the distribution of percent of reachable goals. When using TC-manipulability, we use the set of initial configurations with the highest TC-manipulability. The results of the comparison shows that using multiple initial configurations and using TC-manipulability as selection criteria improves performance. Because of the limitations of our TC-manipulability measure (described in section V-A), TCS did not necessarily select initial configurations with the highest performance. However, they were consistently above the average when not using TC-manipulability and with lower standard deviation. We performed Wilcoxon Rank Sum tests between each method shown in the table and found $p < 0.01$ for all comparisons except the bathing task, which had $p = 0.055$ between the with and without TC-manipulability conditions. Without using TC-manipulability, there is a risk of making a poor selection for the initial configuration.

Figure 6 shows the initial configurations selected by TCS for each task, based solely on TC-manipulability and TC-reachability. These results assume that there is no cost associated with the robot moving to the initial configurations.

Figure 1 shows an interesting solution; when trying to shave a user in a configurable bed, the robot can reach shaving poses well from behind the bed with the head rest lying flat. In [1], the researchers were able to find a good set of two initial configurations to enable a PR2 to help a person with severe motor impairments shave, but they had extensive experience with robots and selecting initial configurations, used time consuming trial and error, and lacked assurances about the availability of alternative solutions.

The TCS solution to shaving in the configurable bed without a backing walls illustrates a potential for TCS to assist in the design of environments. By relaxing constraints on the problem (e.g. removing the wall), it can find interesting solutions to tasks. It may be desirable to design the environment to allow such solutions. Without a wall behind the bed, the PR2 is able to perform the shaving task from a single initial configuration.

A. Limitations

There are various limitations to TCS and the evaluations we performed.

We hand designed the task models for this paper, and have not evaluated how well they actually represent the tasks. In addition, all tasks in this work were defined with full 6-DoF goal poses, although some tasks, such as sponge baths, have position requirements and few orientation requirements. In this work we assume tasks must be performed collision-free, but in [2], we found that contact is both allowable and useful for increasing the workspace in assistive tasks. We also did not address tasks with complex motions like dressing, or tasks with high strength requirements like lifting or ambulating. A valuable step would be to evaluate the performance of this system with a real PR2 and representative users.

The performance of TCS depends on the discretization, which introduces error and a potential to miss solutions. In-

TABLE III: Evaluation of performance of TCS on various tasks. Shows the effect on performance of selecting initial configurations with and without using TC-manipulability and limiting to 1 or 2 initial configurations in a solution set. Differences are statistically significant ($p < 0.01$ in Wilcoxon Rank Sum tests) between each comparison shown except the bathing task between with and without using TC-manipulability (where p=0.055). Feeding in wheelchair saw no increase in TC-manipulability from 2 initial configuration.

		TC-Manipulability	
Task	Number of Initial Configurations	Without: mean (std)	With: mean (std)
Shaving: wheelchair	1	66.7% (13.0)	70.4% (6.1)
	2	94.0% (11.6)	99.9% (1.6)
Feeding: wheelchair	1	87.5% (25.3)	95.5% (14.4)
	2	N/A	N/A
Brushing: wheelchair	1	92.6% (10.8)	98.9% (4.5)
	2	99.7% (2.4)	100.0% (0.0)
Shaving: bed	1	67.0% (8.8)	69.0% (7.3)
	2	95.9% (8.5)	99.9% (1.4)
Bathing: bed	1	68.4% (9.2)	70.6% (4.9)
	2	79.0% (7.9)	79.5% (6.5)

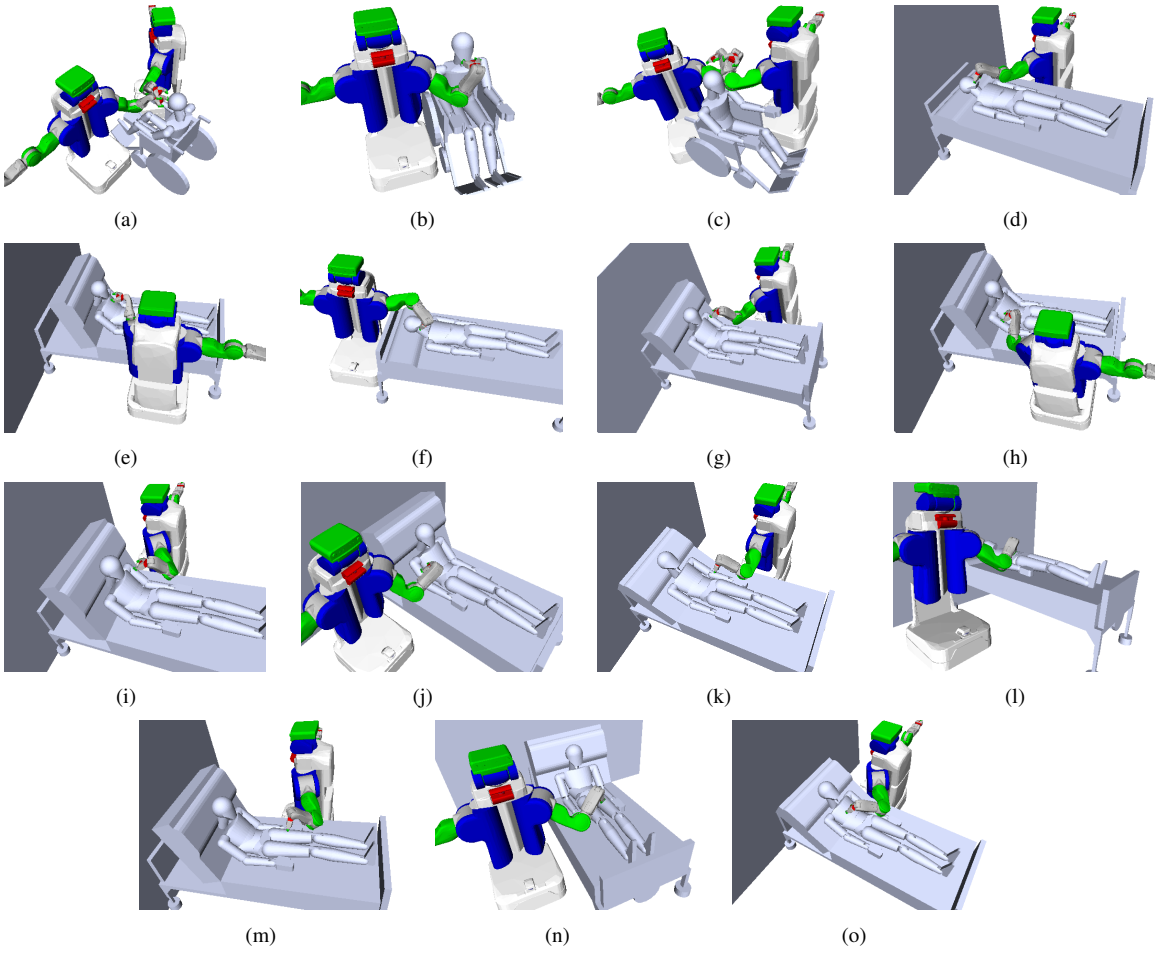


Fig. 6: Visualization of the initial configuration solution with the highest TC-manipulability score for each task. Some task solutions are shown in two images. The images show (a) shaving (b) feeding (c) brushing (d) shaving in bed config #1 (e) shaving in bed config #2 (f) shaving in bed without wall (g) bathing config #1 (h) bathing config #2 (i) scratching left upper arm (j) scratching right upper arm (k) scratching left forearm (l) scratching right forearm (m) scratching left thigh (n) scratching right thigh (o) scratching chest

creases to the discretization space and resolution are limited by computation costs for scoring and the required memory for saving and loading scores.

We found that with our discretization resolution and maximum cardinality of the set of initial configurations, the number of poses for a task and the difficulty of the task (no single initial configuration that can reach all poses) were directly related to the size of the saved files. Scratching task

save files were all less than 3 MB, but the bathing in bed task generated 5 files averaging 77 MB. We used Python to generate the score files and joblib to save them; alternative methods may be able to generate scores faster and store them in smaller files.

Although we have shown that our TC-manipulability score can be used to select good initial configurations, it has some limitations. Kinematic isotropy, the foundation of our TC-manipulabilty score, does not account for joint limits or environmental constraints. In the scratching chest and scratching left upper arm tasks, the initial configuration of the robot put the arm close to its joint limits. There may be value in preferring joint configurations away from joint limits and obstacles, potentially with a weighting function as in [25].

VI. Conclusion

In this work we have presented Task-centric initial Configuration Selection (TCS), a method that uses a measure of task-centric manipulability to accommodate for pose estimation error, considers various environmental degrees of freedom, and can determine a set of initial configurations from which a robot can perform a task. The system can select in real-time an initial configuration solution for the robot to perform the task by precomputing TC-reachability and TC-manipulability scores for sets of configurations. We created 11 models of ADL tasks to test our system in simulation and showed that TCS chooses an effective set of initial configurations for each tasks and could find a configuration consisting of 4 DoF for a robot's mobile base and 4 DoF for a configurable bed.

***Acknowledgment**: This work was supported in part by the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR), grant 90RE5016-01-00 via RERC TechSAge, and by NSF Award IIS-1150157. We thank Henry and Jane Evans for providing input on assistive tasks and actuated beds. We also thank Kevin Chow and Connor Eaton for making the human model used in this work.*

References

[1] K. P. Hawkins, C.-H. King, T. L. Chen, and C. C. Kemp, "Informing assistive robots with models of contact forces from able-bodied face wiping and shaving," in *RO-MAN, 2012 IEEE*. IEEE, 2012, pp. 251–258.

[2] P. M. Grice, M. D. Killpack, A. Jain, S. Vaish, J. Hawke, and C. C. Kemp, "Whole-arm tactile sensing for beneficial and acceptable contact during robotic assistance," in *Rehabilitation Robotics (ICORR), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–8.

[3] D. Park, A. Kapusta, Y. K. Kim, J. M. Rehg, and C. C. Kemp, "Learning to reach into the unknown: Selecting initial conditions when reaching in clutter," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 630–637.

[4] J.-O. Kim and P. Khosla, "Dexterity measures for design and control of manipulators," in *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, Nov 1991, pp. 758–763 vol.2.

[5] M. L. Walters, M. A. Oskoei, D. S. Syrdal, and K. Dautenhahn, "A long-term human-robot proxemic study," in *RO-MAN, 2011 IEEE*. IEEE, 2011, pp. 137–142.

[6] M. L. Walters, K. Dautenhahn, R. Te Boekhorst, K. L. Koay, D. S. Syrdal, and C. L. Nehaniv, "An empirical framework for human-robot proxemics," *Procs of New Frontiers in Human-Robot Interaction*, 2009.

[7] J. Mumm and B. Mutlu, "Human-robot proxemics: physical and psychological distancing in human-robot interaction," in *Proceedings of the 6th international conference on Human-robot interaction*. ACM, 2011, pp. 331–338.

[8] L. Takayama and C. Pantofaru, "Influences on proxemic behaviors in human-robot interaction," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 5495–5502.

[9] M. L. Walters, K. Dautenhahn, R. Te Boekhorst, K. L. Koay, C. Kaouri, S. Woods, C. Nehaniv, D. Lee, and I. Werry, "The influence of subjects personality traits on personal spatial zones in a human-robot interaction experiment," in *Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on*. IEEE, 2005, pp. 347–352.

[10] E. A. Sisbot, L. F. Marin-Urias, X. Broquere, D. Sidobre, and R. Alami, "Synthesizing robot motions adapted to human presence," *International Journal of Social Robotics*, vol. 2, no. 3, pp. 329–343, 2010.

[11] J. Mainprice, E. A. Sisbot, L. Jaillet, J. Cortes, R. Alami, and T. Simeon, "Planning human-aware motions using a sampling-based costmap planner," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5012–5017.

[12] E. A. Sisbot, L. F. Marin, R. Alami, and T. Simeon, "A mobile robot that performs human acceptable motions," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006, pp. 1811–1816.

[13] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon, "A human aware mobile robot motion planner," *Robotics, IEEE Transactions on*, vol. 23, no. 5, pp. 874–883, 2007.

[14] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013.

[15] B. Redmond, R. Aina, T. Gorti, and B. Hannaford, "Haptic characteristics of some activities of daily living," in *Haptics Symposium, 2010 IEEE*. IEEE, 2010, pp. 71–76.

[16] A. Jain and C. C. Kemp, "Improving robot manipulation with data-driven object-centric models of everyday forces," *Autonomous Robots*, vol. 35, no. 2-3, pp. 143–159, 2013.

[17] D. Hsu, J.-C. Latcombe, and S. Sorkin, "Placing a robot manipulator amid obstacles for optimized execution," in *Assembly and Task Planning, 1999.(ISATP'99) Proceedings of the 1999 IEEE International Symposium on*. IEEE, 1999, pp. 280–285.

[18] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: representing robot capabilities," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. Ieee, 2007, pp. 3229–3236.

[19] F. Zacharias, W. Sepp, C. Borst, and G. Hirzinger, "Using a model of the reachable workspace to position mobile manipulators for 3-d trajectories," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*. IEEE, 2009, pp. 55–61.

[20] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, "Reachability and capability analysis for manipulation tasks," in *ROBOT2013: First Iberian Robotics Conference*. Springer, 2014, pp. 703–718.

[21] D. Leidner, A. Dietrich, F. Schmidt, C. Borst, and A. Albu-Schaffer, "Object-centered hybrid reasoning for whole-body mobile manipulation," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1828–1835.

[22] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1970–1975.

[23] F. Stulp, A. Fedrizzi, and M. Beetz, "Learning and performing place-based mobile manipulation," in *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*. IEEE, 2009, pp. 1–7.

[24] A. R. Tilley, *The Measure of Man and Woman: Human Factors in Design*. New York: Wiley, 2002.

[25] F. L. Hammond, "Synthesis of k th order fault-tolerant kinematically redundant manipulator designs using relative kinematic isotropy," *International Journal of Adaptive and Innovative Systems*, vol. 2, no. 1, pp. 73–96, 2014.