Task-oriented Motion Mapping on Robots of Various Configuration using Body Role Division

Kazuhiro Sasabuchi, Naoki Wake, and Katsushi Ikeuchi

Abstract—Many works in robot teaching either focus on teaching a high-level abstract knowledge such as task constraints, or low-level concrete knowledge such as the motion for accomplishing a task. However, we show that both high-level and low-level knowledge is required for teaching a complex task sequence such as opening and holding a fridge with one arm while reaching inside with the other. In this paper, we propose a body role division approach, which maps both high-level task goals and low-level motion obtained through human demonstration, to robots of various configurations. The method is inspired by facts on human body motion, and uses a body structural analogy to decompose a robot’s body configuration into different roles: body parts that are dominant for achieving a demonstrated motion, and body parts that are substitutional for adjusting the motion to achieve an instructed task goal. Our results show that our method scales to robots of different number of arm links, and that both high and low level knowledge is mapped to achieve a multi-step dual arm manipulation task. In addition, our results indicate that when either the high or low level knowledge of the task is missing, or when mapping is done without the role division, a robot fails to open a fridge door or is not able to navigate its footprint appropriately for an upcoming task. We show that such results not only apply to human-shaped robots with two link arms, but to robots with less degrees of freedom such as a one link armed robot.

I. INTRODUCTION

In robot teaching, one way to teach a manipulation task is to provide the high-level knowledge of the geometric constraints required for the task. However, when a task is part of a longer multi-step task sequence, this high-level approach lacks important information on how to complete the task in a way that takes into account the entire sequence. For example, when picking up an item from a closed cabinet, a robot must first reach the door handle position of the cabinet, but do so in a way that enables achieving two following tasks: opening the cabinet, and then reach inside the cabinet with another arm. Planning a motion with the entire task sequence in mind is essential, but doing so with a general motion planner requires a very complex modeling of the task sequence.

Meanwhile, humans are able to move their body with the entire sequence in mind, thus, mimicking a human motion will provide us hints on how to motion plan on a robot. The question is, how to integrate this low-level motion information with the high-level task constraints.

We approach this integration problem by mimicking only the human arm motion instead of the whole body motion, and by dividing the robot configuration into two groups: a group corresponding to the human arm which follows the human arm motion, and a remaining group that solves the task constraints. The reason we focus on the arm motion and use such grouping, is that, the strategy allows us to use the structural analogy of the human body motion: the human arm has control over the motion, and the human trunk acts as a range substitution [1]. Since the arm is the most dominant part of the human body motion, the arm motion should provide us many valuable hints for accomplishing a task, including collision avoidance but also indirect control of the robot’s base positioning in a multi-step task sequence.

Our main contributions in this paper are: First, we present a method that maps both high-level task constraints and low-level motion information using the structural analogy of the human body motion; and show how the method is used with data obtained from an actual sensing system [2]. Second, we present how the mapping method scales to robots of various configuration; including robots with less, equal, and more arm links compared to the human arm. Third, we prove the effectiveness of mapping human arm motion in tasks that must take into account the entire task sequence; but also show the effectiveness on robots that have a slightly different body configurations from the human body.

II. RELATED WORKS

Robot teaching is a wide area of research. Low-level approaches include a kinesthetic demonstration [3] which...
teach by passively moving the robot’s joints, or a direct joint-to-joint mapping of the human body motion [4]. However, kinesthetic demonstrations are difficult to scale for robots with high degrees of freedom (DoF) or when manipulation requires a mobile base movement. In addition, these approaches obtain task information indirectly by repeating the demonstration several times [5], or obtain task information only through correction [6]. The mapping does not scale to robots with different body configurations.

In contrast, high-level approaches rather focus on the symbolic or geometric constraints of the task. Early works have focused on object state transitions [7], and recent works combine geometric constraints with keyframes in a multi-step task demonstration [8]. Many of these approaches focus on the end-effector movements and assume that the task constraints can be solved by a general motion planner. However, this is only true when the robot has enough reachability, which is not the case with domestic robots with a compact structure [9]. An appropriate arm motion integrated with full body positioning is essential for achieving a complex task sequence with such domestic robots.

Few works have tackled the problem of mapping or automatically achieving mobile base movement from demonstration. Strake et al. approached the problem by mapping the human torso movement for a single arm task [10].

Early works that integrate both high-level constraints and low-level motion focused mainly on grasping [11]. Meanwhile, works that focus on full body motion is found in computer graphics [12]. However, the main goals in computer graphics was to solve the constraints between the agent and the floor, and not interaction with articulated objects. Recent works integrate high-level task knowledge and low-level arm motion in a manipulation context by leveraging human played (operated) robot motion and then performing zero-shot teaching through high-level verbal task instructions [13]. Yet, such leveraging by operation is difficult with tasks requiring base positioning or multiple arm manipulation. For these difficult cases, we must leverage knowledge from human motion demonstration.

III. PROBLEM DEFINITION

In our problem, we assume that an instruction (a sequence of tasks) is converted to a list of task constraints, and the corresponding human arm motion to achieve the task is demonstrated along with the instructions (Figure 1). Below we explain the task constraints and the arm motion we use for planning the robot motion.

A. Task Constraints

For the task constraint, we consider a desired position-force goal state \((p_f, f)\) of the robot’s end-effector, where the position goal state is the state achieved by an action \(\Delta p\) of the end-effector in a motion free direction, and the force goal state is the state achieved by an action \(\Delta f\) of the end-effector in a non-motion free direction (the direction the end-effector is in contact with a non-deformable rigid surface). Many tasks contacting or manipulating a rigid object can be described as a list of this position-force goal state representation (e.g., pick-and-place, door opening, pressing a button, operating a kitchen faucet, etc.) Below we provide some concrete examples (Figure 2).

- Carrying an object: \([(p_0, f_0), (p_0 + \Delta p, f_0)]\) where \(p_0\) is the start position of the end-effector moved by \(\Delta p\) to a desired goal position.
- Lifting from a table: \([(p_0, f_0), (p_0, f_0 + \Delta f)]\) where \(f_0\) is the starting force of the end-effector increased by \(\Delta f\) to a desired goal force that is able to counter the attaching force (gravity, magnetic, etc.) on an object.
- Wiping a table: \([(p_0, f_0), (p_0 + \Delta p, f_0 + \Delta f)]\) where \(p_0\) is the start position moved to a desired goal position on a table, and \(f_0\) is the starting force changed to a desired goal force for ensuring the table is wiped.

Of the end-effector actions \(\Delta p\) and \(\Delta f\), \(\Delta f\) acts on a direction in contact with a rigid surface and is constrained; thus the change in motion produced by \(\Delta f\) is very slight. As a result, achieving the force constraint usually does not conflict with mimicking the demonstrated arm motions. Meanwhile, \(\Delta p\) acts on a motion free direction and may drastically change the robot’s arm configuration from a mimicked motion. In the remainder of the paper, we will focus mainly on achieving the desired position goal state (task position goal) beside mimicking a demonstrated motion.

One other constraint to note, is the orientation of the end-effector when performing the end-effector actions (orientation goal). The orientation goal is usually defined from the properties (e.g., the shape) of a manipulating object, but must sometimes be obtained from the human demonstration to consider the entire task sequence. An example case is when determining to grasp a can drink from the side or top in a picking task. Side grasping is appropriate when placing the can drink inside a shelf, whereas top grasping is appropriate when placing inside a basket. Which grasp to use depends on the properties of the placing location, which is not a direct task constraint of the grasping task, but rather an information from the entire task sequence. We will assume that, whether to obtain the orientation goal from the object property or human demonstration is defined a priori in a database about the manipulating object.
B. Human Arm Motion

For the arm motion to mimic, we consider a list of desired arm postures (arm posture goal) represented in some intermediate representation, such as the name of the posture. In order to cover a variety of arm postures, we name a motion for every possible combination of the human upper and forearm pointing directions in some digitalized direction space (Figure 3). In this paper, we will use a direction space used in existing human motion representations [14][15], where the direction is divided into eight horizontal directions (forward, left forward, left, ...) and five vertical directions (south pole, low, middle, high, north pole). To our survey, when limited to in-front single arm manipulations, the number of valid direction combinations of the human upper arm and forearm (the number of named postures) is 79 using this eight-by-five digitalization.

Another way to represent an arm posture is to represent the pointing direction of the upper arm and forearm each as a vector of continuous float values. However, a continuous data representation will contain many noises in the raw data and will require a well-defined set of equations to convert to a robot motion. Meanwhile, an intermediate representation allows us to define a finite number of mapped configurations a priori, and is able to filter noisy jumps or obvious detection errors in the human motion (e.g., unnatural arm twisted postures can be checked a priori in a digitalized representation and then be defined as unacceptable).

IV. BODY ROLE DIVISION METHOD

In this section, we explain our method for planning a robot motion that satisfies both the task constraints and the mimicking of the human arm motion. By mimicking the human arm motion, we expect to achieve a robot motion with the entire task sequence in mind.

As explained in Section III, we divide the robot configuration \( \mathbf{q} \) into two groups: a configurational group \( \mathbf{q}^c \) which are the joints that map the human arm motion; and a positional group \( \mathbf{q}^p \) which are the remaining joints that solve the task constraints (desired end-effector states). An exceptional constraint that is not solved with the positional group is the orientation goal from Section III-A. The orientation goal is sometimes obtained through human mimicking, thus, the group solving the goal (the orientational group) should be a subset of the configurational group.

The configurational group corresponds to the arm-to-hand (upper arm, forearm, and wrist-to-hand) on the human body. In this paper we will consider robots that have an arm attached to some base, and an end-effector attached to the end of the arm. In most cases, the configurational group is the arm and the end-effector.

The positional group corresponds to the trunk of the human body, which is the waist or torso of the robot, but since simple-structured robots may not have such structure (or not enough joint range as the human), we will also include the robot’s base. Note that, the base movement does not correspond to the human footsteps, but is positioned to substitute the arm movement (Section IV-A.2). To integrate mobile base movement, we will consider base movements as part of the robot joint configuration by defining a virtual prismatic and/or revolute joint attached to the robots base.

The orientational group corresponds to the wrist-to-hand part of the human arm. This is usually the wrist and end-effector on the robot, which indeed by itself is able to solve the orientation goal [16]. This subset of the configurational group is required due to the characteristics of the orientation goal, but structural facts on the human motion also insist that the human wrist motion is independent from the upper and forearm motion [17]; thus defining a subset is also reasonable from a structural analogy perspective.

Using this idea of body role division which decomposes the robot body into a configurational group, positional group, and an orientational group, we solve the arm posture goal, task position goal, and orientation goal from Section III. For simplicity, we will begin with the case of a single arm posture goal, a single task position goal, and a single orientation goal (e.g., the moment of grasping). Our method uses a step-by-step calculation on each role group, which is described below:

1) Map the arm posture goal to a mapped configuration \( \mathbf{q}_0^c \) which define a set of joint values for the configurational group. Set some predefined default configuration \( \mathbf{q}_0^p \) (e.g., zero values) for the positional group.
2) By changing the joint values in the orientational group, modify \( \mathbf{q}_0^c \) to joint configuration \( \mathbf{q}_1^c \) which satisfies the orientational goal \( \Omega_{goal} \).
3) Find a final configuration \( \mathbf{q} \) which satisfies the task position goal \( \Omega_{goal} \) by mainly changing the joint values in the positional group, but also by making sure that the configurational group is maintained using a configuration constraint \( \Omega_{cons} \), and a group connection constraint \( \Omega_{cons} \).

The search of a configuration in the last two steps can be done by applying the goals and constraints as a fitness function in a genetic algorithm [18]. We explain the details of each step in each of the below subsections.
A. Mapping the Arm Posture Goal

The mapping design of a named human arm posture to a mapped configuration $q_0^c$ depends on the number of links (excluding the end-effector) that compose the robot arm. We define three patterns (Figure 4): the equal degrees of freedom (DoF) case where there are exactly two links (same as the human demonstrator), the less DoF case where there is only one link, and the more DoF case where there are more than two links. We will assume that for the equal DoF and less DoF, each link is nearly equal to the human arm links.

1) Equal Degrees of Freedom: Since the number of links is equivalent, a naïve mapping approach is to copy the named pointing direction of the human upper and forearm, to the upper and lower robot arm link. However, this way of mapping has no information on the joint-level interpolation between two mapped configurations, thus may lack human motion characteristics. For example, let us say an arm is reaching straight from a bent elbow position. The straight arm is a singular point, and depending on the twist amount of the upper arm, different end-effector movements will be generated during the interpolation.

To achieve a smooth interpolating motion or a most-likely collision avoiding motion, we must consider the characteristics of the human arm motion. According to Tadoko et al. [19], the upper arm usually does not twist during a straight reaching motion, but twists when moving the arm to different heights. Therefore, a mapped configuration must be created in a way such that, 1) the pointing direction is kept as much as possible, but 2) the upper arm does not twist between reaching transitions and only twist when there is transition in the height direction. Since we will be using a finite set of arm postures (as explained in Section III-B), the number of transition patterns is also finite. When a robot cannot precisely copy the pointing direction for one of its arm links (e.g., due to joint limitations), we will prioritize the twist constraint when designing the mapped configuration.

2) Less Degrees of Freedom: One approach to map arm postures to robots that have only one arm link is to sum the named pointing direction of the human upper arm and forearm into one direction [15]. However, while this approach may be suitable for gesture motion, manipulation motion have a slightly different characteristic. That is, collision between the forearm and the environment is avoided by the positioning of the elbow and wrist, thus, a summed pointing direction may miss the collision-avoiding essence. To achieve a mapped configuration that is most likely not under collision, we will mainly map the forearm pointing direction to the arm link, and refer the root of the arm link as the elbow. The assumption that lies here is that, the upper arm is mainly used to adjust the forward/outward positioning of the elbow, therefore, such motion essence can be (in most cases) alternatively managed with the positional group. To our survey, only 12 out of the 79 named arm postures lie in an exceptional case requiring an upper arm reach, such as reaching over a table.

To achieve the forearm direction, the arm link must be actuated using a horizontally rotating joint and a vertically rotating joint. For some robots, the horizontal rotation may depend on the rotation of the base, therefore, in addition to the arm link, the virtual base rotation may also be included in the configurational group.

3) More Degrees of Freedom: Since there are more links than required for the mapping, we must choose the links to include in the configurational group. As in the less DoF case, to achieve a most-likely not-under-collision configuration, the arm link should have the same length as the human arm. However, a multi-link arm may be composed of a number of short links [20]. Therefore, we will choose $N$ closest links from the end-effector, and $M$ next-closest links such that the $N, M$ links compose approximately the same length as the human forearm and upper arm respectively. If $M$ is not long enough to compose an upper arm, we will treat the arm as the same as the less DoF case, otherwise, we will treat the $N + M$ links as the same as the equal DoF case.

B. Solving the Orientation Goal

In order to represent an orientation goal in relation to human mimicking, we use the pointing direction of the palm [14]. Using this palm analogy, we define a fixed palm unit vector $v_p$ on the robot’s end-effector $E$ represented in the $E$ coordinate (Figure 5). The orientation goal is then to point this palm vector toward a desired direction $v^\text{goal}_p$ in some fixed task coordinate. With only this condition, the end-effector may take any rotated pose around the palm vector. Therefore, we may choose one fixed perpendicular unit vector $v_n$ represented in the $E$ coordinate, and make sure $v_n$ points to some desired direction $v^\text{goal}_n$ in the task coordinate.

![Figure 5](image_url)

Fig. 5. Figure explaining the orientation goal, which uses the palm direction representation taken from the human motion analogy.
An example of $v_p^{goal}$ is a demonstrated direction such as grasping a can from the side or top. An example of $v_n^{goal}$ is a constrained direction parallel to the axis of a cylindrical handle.

Let $R_q$ be a coordinate transformation matrix that transforms $v_p$, $v_n$ in the $E$ coordinate to the task coordinate when the robot’s configuration is $q$. Then, using a threshold $\theta_p$, $\theta_n$ the orientation goal is written as below:

$$\Omega_{ogal}(q) : \left\{ \begin{array}{ll} 1 - v_p^{goal} : R_q v_p < \theta_p \\
1 - v_n^{goal} : R_q v_n < \theta_n \end{array} \right. \quad (1)$$

C. Solving the Task Position Goal

Let $p$ be the desired position of the end-effector in the task coordinate, and $h(q_s)$ the end-effector position when the robot’s configuration is a sampled configuration $q_s$ (calculated using forward kinematics). Then, using a threshold $d$ the task position goal is written as below:

$$\Omega_{pgal}(q_s) : \|h(q_s) - p\| < d \quad (2)$$

We apply two constraints while solving this task position goal. One is a configuration constraint which ensures that the joint values of the configurational group is kept near the values of the mapped configuration from step 1 and 2. The other is a group connection constraint: when the links actuated by the positional group are the parent or child of the configurational group, a change in value in the positional group may change the look of the links (pointing directions) actuated by the configurational group. The group connection constraint ensures that such situation is avoided.

1) configuration constraint: Let $q_s^c = \{q_s^c_i| i = 1, \ldots\}$ be the configurational group of a sampled configuration and $q_s^c_i$ be the $i$-th joint value. Let $q_1^c = \{q_1^c_i| i = 1,\ldots\}$ be the configuration solved in step 2, and $d_c$ some defined threshold. Then, the configuration constraint is written as below:

$$\Omega_{con}(q_s^c) : \sum_i |q_s^c_i - q_1^c_i| < d_c \quad (3)$$

2) group connection constraint: One way to solve the group connection constraint is to use a similar strategy as $\Omega_{con}$. Let $L$ be a subset of the positional group that influences the look of the links actuated by the configurational group. Let $q_0^L \in q_0^P$ be a partial configuration of the joint configuration from step 1. The subset positional group in a sampled configuration $q_s^L = \{q_s^L_i| i = 1, \ldots\}$ is kept close to $q_0^L = \{q_0^L_i| i = 1, \ldots\}$ within a threshold $d_p$ using below:

$$\Omega_{pcon}(q_s^L) : \sum_i |q_s^L_i - q_0^L_i| < d_p \quad (4)$$

V. USING WITH A SENSING SYSTEM

We extend our discussion of applying our body role division method on a single data point to a series of data obtained from a sensing system (Figure 1). A sensing system outputs two types of data, a position data and a posture data. The position data contains a series of desired task position goals. The posture data is a full recording of the demonstrated arm postures (which is then named to represent an arm posture goal).

The position data itself only contains information about the geometric values to solve, but the sensing system looks at when the human hand trajectory visited those values, therefore, is able to pair the task position goal with a corresponding arm posture goal from the posture data (Figure 7). When a robot is manipulating (or is about to manipulate) an object at the paired data point, the orientation goal is extracted from the arm posture (palm direction) or from the geometric data of the manipulating object (which can be searched in a database using the name of the object). When a robot is not manipulating an object, the orientation goal step of the calculation in Section V is skipped.

One thing to note is that, while some task position goals have re-usable values (e.g., manipulating an articulated object fixed to the environment), others may be volatile (e.g., position of a can). We assume that such re-usable/volatile tag is provided by the sensing system. The position values for the volatile points are captured from recognition during robot execution and are overwritten. As a limitation of this paper, we will only consider the case where the positional change of an object is slight (which is the case when picking a mostly-same-place-located item from a fridge). Thus, despite the positional difference, the required posture can be assumed to be re-usable.

Another thing to note is that, the paired data could be
sparse or dense. An example of when a data is sparse, is a pick-and-place data (Figure 9). The data consists of only a few data points on where the robot end-effector should visit. The robot is able to achieve the task as long as the key waypoints detected by the sensing system are visited and collision is avoided. An example of when a data is dense, is a door-opening data. The robot end-effector must follow the exact positions on a specified door-opening trajectory.

When the data is dense, directly using the posture name of a digitalized posture as the arm posture goal, will have an issue. The posture rarely changes with this naming strategy (Figure 7) and may generate sudden jumps between postures. To prevent such issues, we represent the arm posture goal in a dense data, with the name of the starting posture, ending posture, and an interpolation parameter $t$. To obtain the mapped configuration, the start and end posture is first mapped to configurations $q(a)^c$, $q(b)^c$, and an interpolated configuration is calculated as $(1-t)q(a)^c + tq(b)^c$.

VI. EXPERIMENT
A. Setup of Role Division in Dual Arm Manipulation

For evaluating our method, we set up a "pick-from-fridge" experiment in a virtual simulator using ROS RVIZ and fake controllers provided by MoveIt. All base movement errors were ignored in this condition. The "pick-from-fridge" consisted three tasks: "T1: reach for the fridge handle" followed by "T2: open the fridge" and "T3: pick a can from inside the fridge." The fridge automatically closes if not held, thus, a dual arm manipulation was required for T3. The geometric model parameters of the fridge and the can were assumed to be known.

We used the SEED-noid robot [21], which has two 7 DoF arms, 2 DoF in the waist, and an equal number of arm links as the human arm. The robot is able to localize itself in the task coordinate using a base laser scan. Bio-IK [18] was used for the inverse kinematics solver.

Following the explanation in Section IV the task position goals for T1 and T3 were represented using visiting points. The task position goals for T2 was represented as a trajectory divided into waypoints-to-follow for every 0.1 [radian] opening of the door. A value smaller than this value will result to base movements less than 3 [centimeters], which is too small and cannot be achieved with the SEED-noid's wheel configurations.

The arm posture goals were obtained from a sensing system for all three tasks, and an interpolation parameter as described in Section IV was used for T2.

Following the notations in Section IV-B, the orientation goal for T1 and T2 used a direction within 45 degrees of the direction perpendicular to the door plane for $v_p^\text{goal}$, and a direction parallel to the door handle axis for $v_n^\text{goal}$. For T3, a demonstrated approach direction was used for $v_p^\text{goal}$, and a direction parallel to the can’s axis was used for $v_n^\text{goal}$.

We compared our method with three other methods described below:

- **low-only** - Apply only the arm posture goal. The initial base position was positioned so that the position of the reached arm and the fridge door handle was matched at the beginning of T2.

- **no-role** - Apply the task position goals and orientation goals, but provide the arm posture goal as an initial seed for solving the inverse kinematics. Other conditions were the same as the high-only condition. Information on the joint groups from Section IV was not provided, therefore, the solver had no idea of which joints can change its values drastically and which joints cannot.

- **proposed** - The proposed body role division method explained in Section IV. Since the real SEED-noid cannot move its base and joints at once due to a power supply restriction, the positional group was used in the following way: the task position goal was solved using the waist under constraint $\Omega_{\text{cons.}}$. When a valid real joint configuration was not found, the base movement was used to solve the position displacement in the same way as the high-only method.

In all four methods, any base movement was not used for T3. Simply applying a positional displacement of one arm (e.g., picking the can) will violate the constrained position of the other arm (e.g., holding the door handle). A task position goal to keep the position of the other arm was added in T3, and the goal was solved by using only the robot’s real joint configurations.

B. Results of Role Division in Dual Arm Manipulation

| Proposed Method | low-only | high-only | no-role | succeeded |
|-----------------|----------|-----------|---------|-----------|
| T2 w/ real joints | 59% | - | 76% | 89% |
| motion jumps | yes | no | yes | no |
| collisions | no | yes | no | no |
| task achievement | failed | failed | failed | succeeded |

Table I shows the results obtained by the four methods. From the table, only the proposed role division method successfully achieved the picking of the can (Figure 8). All other methods failed to find an inverse kinematics solution to picking the can from the final base position achieved after opening the door. In addition to not being able to complete the full task sequence, the high-only and no-role method had a motion jump where the robot hand departed from the door handle (Figure 10A,C middle). This would break the door handle if executed on a real robot. The low-only method had
Fig. 8. Opening the fridge with one arm and picking a can inside the fridge with the other arm, using our proposed body role division method. On the top row, the first three pictures indicate the first half of opening the fridge before moving the base, the next three pictures indicate the second half after moving the base, the last three pictures indicate picking the can. The bottom row is the execution with a real robot.

Fig. 9. Our method on a less DoF robot (top row). The bottom row compares our method (right) with using only the task position goals (left), from a bird-eye view. Our method is able to look inside the fridge from a closer and easier to see position, including seeing the back of the fridge door.

Fig. 10. (A) Start of T2, motion jump, end of T2 in the high-only method. (B) Start of T2, a motion colliding with the door due to deviation from trajectory, end of T2 in the low-only method. (C) Start of T2, motion jump, end of T2 in the no-role method. The red and orange circles in (A) and (C) indicate the position of the departing robot hand and the handle respectively.

an apparent collision with the door during motion execution (Figure 10B). With the proposed method, there were no collisions in all tasks including the can picking. This was due to the careful design in mapping (Section IV-A.1).

The cause of the different results between high-only, no-role, and proposed which all consider the task position goals, can be explained for the following two reasons: First, by defining an orientational group, we are able to solve the orientation goal as an individual step. With this step-by-step calculation, the inverse kinematics solver finds an acceptable (not exact) orientation goal that is constrained under a desired configuration. However, when all goals are solved at once (without steps), the solver gets stuck to a local minimum that satisfies the exact orientation goal, but does not maintain the desired configuration (even if used as an initial seed). With the structure of the SEED-noid robot, this results to an awkward twisting configuration. Second, by dividing the joints contributing to the arm posture and the joints contributing to the task goal, we are able to define a metric (equation 3) for deciding whether a configuration deviates from the demonstrated motion. By making sure there is no deviation (and since motion continuity between likely transitions is guaranteed by the mapping scheme), we are able to avoid jumping configurations. Moreover, the results show that a demonstrated arm motion is able to indirectly guide an appropriate base positioning, which accomplishes the multi-step task sequence of picking a can from the fridge.

Further looking at the results, the first row of the table sums the percentage of waypoints that did not require base movement (succeeded in solving with the real joints) in T2. We see that the more we consider the mapping of the demonstrated arm motion, the less we fail with solving using the real joints. The result insists that the low-level arm motion knowledge indeed provide valuable information about how to execute a task in a successful way.

C. Role Division on Robots with Less Degrees of Freedom

We also evaluated the fridge task with the HSR robot [9], which has less DoF than the human arm. Since the HSR only has one arm, we assumed tasks T1, T2 followed by
“T3’: look for a can inside the fridge.” The experimental conditions (including the posture data) were the same as the SEED-noid except the height of the fridge was adjusted to meet an operation-possible height of the HSR, and (due to HSR’s simple structure) an analytic inverse kinematics solver was used in the high-only method. Unlike the SEED-noid, the HSR had no limitations for moving the base and joints at once, thus, the body role division explained in Section IV.A.2 was used without any robot-specific adjustments.

A comparison of the high-only and proposed method is shown in Figure 9. Since the HSR uses an analytic inverse kinematics solver, the solution provided by the high-only method is not as awkward as the SEED-noid. However, the final base positioning is different between the two methods. Our proposed method allows the robot to look inside the fridge from a close and in-front position (and also the back of the door, where items could be stored in an actual fridge), whereas, the high-only results to a far and slightly-to-the-side position and cannot see the back of the door. Thus, the human arm motion provide valuable information for accomplishing the entire task, even for a simple robot such as the HSR.

VII. CONCLUSIONS

Applying both high-level task constraints and low-level motion knowledge is essential for solving a complex task sequence. The task constraints lack information on how to execute a task with the entire task sequence in mind. The motion knowledge lack information on how to scale the motion in a way that achieves the task objective. The two different level of information about a task is applied by dividing a robot’s body configuration into configurational, positional, and orientational groups. Compared to only providing the task constraints, our method provided more stable results, was 30% more successful in solving the door opening task using the robot’s joint configurations, and was the only method that completed the dual arm picking with a robot arm almost the same length as the human. It was clear that by using arm motion information, our method indirectly prepares the robot’s footprint for an upcoming task in a task sequence. This not only applied to robots with similar structure as the human, but also with robots with simpler structure such as a one-link arm.

Lastly, there are a few ways to extend our work. One direction is to consider environments with dynamic changes, where location of objects and required motion is completely different from demonstration. In such situations, we may collect a motion dataset beforehand, then retrieve from the dataset the most likely motion tied to a particular situation. The tied motion can then be used as the posture goal in our method. A second direction of extension is to consider tasks that require specific leg motion. Tasks such as climbing a stair with a biped robot while holding on a stair rail, may require a different role division, or at least a different decision on which body motion is dominant and which are substitutional.

REFERENCES

[1] TR Kaminski, C Bock, and AM Gentile. The coordination between trunk and arm motion during pointing movements. Experimental brain research, 106(3):457–466, 1995.
[2] Naoki Wake, Iori Yanokura, Kazuhiro Sasabuchi, and Katsushi Ikeuchi. Verbal-based focus-of-attention for learning-from-demonstration. 2020.
[3] Haimena D Argulli, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. Robotics and autonomous systems, 57(5):469–483, 2009.
[4] Behzad Dariush, Michael Gienger, Arjun Arumbakkam, Youding Zhu, Bing Jian, Kikuo Fujimura, and Christian Goerick. Online transfer of human motion to humanoids. International Journal of Humanoid Robotics, 6(02):265–289, 2009.
[5] Anude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. Survey: Robot programming by demonstration. Handbook of robotics, 59(BOOK_CHAP), 2008.
[6] Andrea Bajcsy, Dylan P Losey, Marcia K O’Malley, and Anca D Dragan. Learning from physical human corrections, one feature at a time. In Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction, pages 141–149. ACM, 2018.
[7] Katsushi Ikeuchi and Takashi Suehiro. Toward an assembly plan from observation. i. task recognition with polyhedral objects. IEEE transactions on robotics and automation, 10(3):368–385, 1994.
[8] Claudia Pérez-D’Arpino and Julie A Shah. C-learn: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 4058–4065. IEEE, 2017.
[9] Takashi Yamamoto, Koji Terada, Akiyoshi Ochiai, Fuminori Saito, Yoshiaki Ashahra, and Kazuto Murase. Development of human support robot as the research platform of a domestic mobile manipulator. ROBOMECH journal, 6(1):4, 2019.
[10] Tim Welschelder, Christian Dornhege, and Wolfram Burgard. Learning mobile manipulation actions from human demonstrations. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3196–3201. IEEE, 2017.
[11] Sing B. Kang and Katsushi Ikeuchi. Toward automatic robot instruction from perception-mapping human grasps to manipulator grasps. IEEE transactions on robotics and automation, 13(1):81–95, 1997.
[12] Katsu Yaman, Yuka Arika, and Jessica Hodgins. Animating non-humanoid characters with human motion data. In Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pages 169–178, 2010.
[13] Corey Lynch and Pierre Seremanet. Grounding language in play. arXiv preprint arXiv:2005.07648, 2020.
[14] Ann Hutchinson Guest. Labanotation: the system of analyzing and recording movement. Routledge, 2014.
[15] Katsushi Ikeuchi, Zhaoqian Ma, Zengqiang Yan, Shunsuke Kudoh, and Minako Nakamura. Describing upper-body motions based on labanotation for learning-from-observation robots. International Journal of Computer Vision, 126(12):1415–1429, 2019.
[16] Girish K Singh and Jonathan Claassens. An analytical solution for the inverse kinematics of a redundant 7 dof manipulator with link offsets. In Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, pages 2976–2982. IEEE, 2010.
[17] Francesco Lacquaniti and John F Soechting. Coordination of arm and wrist movement during a reaching task. Journal of Neuroscience, 2(4):399–408, 1982.
[18] Sebastian Starke, Norman Hendrich, and Jianwei Zhang. A memetic evolutionary algorithm for real-time articulated kinematic motion. In 2017 IEEE Congress on Evolutionary Computation (CEC), pages 2473–2479. IEEE, 2017.
[19] Satoshi Tadokoro, Makoto Aikawa, and Kazuhiro Ichimichi. An analysis of human upper extremity motions. The Japanese Journal of Ergonomics, 26(1):41–47, 1990.
[20] Zheng Li, Ruxu Du, Man Cheong Le, and Song Mei Yuan. Design and analysis of a biomimetic wire-driven robot arm. In ASME 2011 International Mechanical Engineering Congress and Exposition, pages 191–198. American Society of Mechanical Engineers Digital Collection, 2011.
[21] Kazuhiro Sasabuchi, Hiroaki Yaguchi, Kotaro Nagahama, Shintaro Hori, Hiroto Mizohana, and Masayuki Inaba. The seednoid robot platform: Designing a multi-purpose compact robot from continuous evaluation and lessons from competitions. IEEE Robotics and Automation Letters, 3(4):3983–3990, 2018.