
Article:
Fang, C, Ding, X, Zhou, C orcid.org/0000-0002-6677-0855 et al. (1 more author) (2019) A²ML: A general human-inspired motion language for anthropomorphic arms based on movement primitives. Robotics and Autonomous Systems, 111. pp. 145-161. ISSN 0921-8890

https://doi.org/10.1016/j.robot.2018.10.006

A^{2}ML: A General Human-Inspired Motion Language for Anthropomorphic Arms Based on Movement Primitives ☆

Cheng Fang[a,b,*], Xilun Ding[b], Chengxu Zhou[a], Nikos Tsagarakis[a]

[a]*Humanoids and Human Centered Mechatronics, Istituto Italiano di Tecnologia, via Morego 30, 16163 Genoa, Liguria, Italy*
[b]*Robotics Institute, Beihang University, Xueyuan road 37, 100191 Beijing, China*

Abstract

The recent increasing demands on accomplishing complicated manipulation tasks necessitate the development of effective task-motion planning techniques. To help understand robot movement intention and avoid causing unease or discomfort to nearby humans towards safe human-robot interaction when these tasks are performed in the vicinity of humans by those robot arms that resemble an anthropomorphic arrangement, a dedicated and unified anthropomorphism-aware task-motion planning framework for anthropomorphic arms is at a premium. A general human-inspired four-level Anthropomorphic Arm Motion Language (A^{2}ML) is therefore proposed for the first time to serve as this framework. First, six hypotheses/rules of human arm motion are extracted from the literature in neurophysiological field, which form the basis and guidelines for the design of A^{2}ML. Inspired by these rules, a library of movement primitives and related motion grammar are designed to build the complete motion language. The movement primitives in the library are designed from two different but associated representation spaces of arm configuration: Cartesian-posture-swivel-angle space and human arm triangle space. Since these two spaces can be always recognized for all the anthropomorphic arms, the designed movement primitives and consequent motion language possess favorable generality. Decomposition techniques described by the A^{2}ML grammar are proposed to decompose complicated tasks into movement primitives. Furthermore, a quadratic programming based method and a sampling based method serve as powerful interfaces for transforming the decomposed tasks expressed in A^{2}ML to the specific joint trajectories of different arms. Finally, the generality and advantages of the proposed motion language are validated by extensive simulations and experiments on two different anthropomorphic arms.

Keywords: Anthropomorphic arm, motion language, task-motion planning, movement primitive.

1. Introduction

Recently, the evolution of robotics research has been gradually developing from industrial robotics to service robotics. Particularly, service robots built and controlled following the principle of anthropomorphism, for instance, humanoid robots, have received considerable attention in this trend. On one hand, the utilization of anthropomorphism for safe and friendly physical or social interaction with people can facilitate our understanding of robot behaviors when the robots are integrated into a human-oriented environment [1]. On the other hand, this kind of anthropomorphic robots provide helpful platforms for experimental validation of theories and hypotheses about biological models formulated by neurophysiologists, neuroscientists and psychologists due to the difficulty or impossibility of the experiments conducted on or with human beings [2].

Anthropomorphic arms with the same shoulder-elbow-wrist configuration and number of Degrees of Freedom (DoF) as human arm (7-DoF) are usually favorable and inclinable choices when they are supposed to be desgined as important components of service robots performing manipulation tasks around humans. These physically anthropomorphic designs and consideration are motivated by the reason that these human-like arms can naturally form a foundation for resembling or achieving human motions and behaviors, which help understand robot movement intention, avoid causing unease or discomfort to nearby humans [3], and even express emotions as appropriate feedback or reaction [4]. Moreover, these anthropomorphic arms provide a natural interface for non-skilled users to operate robots [5], and at the same time enable the robots to acquire human manipulation skills through intuitive learning by imitation and demonstration [6, 7].

It can be foreseen and envisaged that service robots will be faced by a wide variety of intricate manipulation tasks when they are inserted into various fields of human society. Therefore, how to plan reasonable motions of their anthropomorphic arms to accomplish these tasks is becoming a very necessary and important issue. In addition, even though similar 7-DoF anthropomorphic arms with the shoulder-elbow-wrist configuration are often mounted on service robots, these arms probably differ in the rotation axis designation, the rotation order of their internal mechanical joints and the link dimension, which are further explained and illustrated in Sections 4.1 and 6.1, re-

☆This work is supported by the European Research Council under Horizon 2020 Programme projects CENTAURO (ICT-23-2014, 644839) and CogIMon (ICT-23-2014, 644727).

*Corresponding author
Email address: cheng.fang@iit.it (Cheng Fang)

spectively. To cope with the diversities of manipulation tasks and anthropomorphic arm platforms, a unified flexible and general task-motion planning framework for this special type of robot arms is highly encouraged to intuitively and uniformly describe and plan their motions to accomplish various and complicated tasks. In addition, to make anthropomorphic arms exhibit human-like motions and behaviors, it is desirable to incorporate anthropomorphism into this framework. Inspired by neurophysiological findings of human arm motion hypotheses/rules and the ideas of modularization and hierarchy of the language formulation and processing in artificial intelligence [8, 9, 10], such a unified task-motion planning framework is therefore developed and shaped into a human-inspired modularized *Anthropomorphic Arm Motion Language* (A^2ML), which constitutes the main contribution of this work.

2. Related work

Researchers initially dealt with the motion planning problem of redundant anthropomorphic arms as a special case of redundancy resolution using local optimization techniques, e.g., *Gradient Projection Method* (GPM) [11]. This type of methods usually work at the velocity level: the linear and/or angular velocties of the end-effector are first planned in Cartesian space. Redundancy resolution approaches are then employed to calculate the corresponding joint velocities to fullfil the planned Cartesian velocities while locally optimizing some criteria with the redundant DoFs. The executable joint trajectories are consequently obtained by integrating these joint velocities. However, these methods were proved to be unpractical for realistic motion planning problems due to several drawbacks. One reason is that they would suffer from singularity problem since they work at the velocity level and usually have to calculate the inverse of the arm kinematic Jacobian. Apart from this aspect, the satisfaction of some hard constraints, such as joint angle/velocity limits and obstacle avoidance can not be always guaranteed in these methods. To overcome these issues, the redundancy resolution problem can be reformulated as a *Quadratic Programming* (QP) based optimization problem, where the error between the expected and resultant Cartesian velocities (calculated by forward Jacobian mapping) is minimized while respecting equality and/or inequality constraints. The QP-based framework can be also extended to a hierarchical structure to realize the control of multiple tasks with different priorities [12, 13, 14, 15]. Another popular category of motion planning methods are the sampling-based motion planning methods, which directly work in the configuration space at the position level. As many feasible configurations as possible subject to the hard constraints are located by random global sampling. Based on these sampled configurations, a graph is then established to descibe the connectivity among them, and serves as a guided map to help the robot to find a feasible path solution from an initial configuration to a specified goal configuration. These methods have been becoming appealing since they excel at solving challenging motion planning problems in high-dimensional complicated configuration space. The typical methods in this category are *Rapidly exploring Random Tree* (RRT) [16] and its variants. Based on a feasible initial solution obtained by the methods above, it can be further refined by employing the covariant gradient optimization technique (CHOMP [17]) or stochastic trajectory optimization strategy (STOMP [18]).

Task planning, which usually involves task decomposition and subtask sequencing [19, 20, 21], is a higher level planning than motion planning for robots to perform more complicated tasks. Many task planners descended from the first robot task planner, STRIPS [22], employed means-ends analysis to guide the robot to move from an initial state to a desired goal-satisfying state following a sequence of actions. However, they usually only took the topological constraints among the states and actions into account without considering actual geometrical cosntraints of robot motions. Therefore, there was an evident gap between task planning and motion planning. How to transform a task to the joint trajectories of a specific robot while being aware of the symbolic and geometrical constraints necessitates an integrated task-motion planning approach.

S. Cambon et al. made pioneering contribution in this direction [23, 24]. A combined treatment of topological and geometrical preconditions and effects of robot actions was proposed in the representation of the task-motion planning problem, where reachability conditions must not be asserted by a task planner but were automatically inferred after checking for the existence of feasible solution(s) using a motion planner. L. P. Kaelbling et al. [25] proposed a task-motion planning framework based on the planning in the belief space of probability distributions over states using hierarchical goal regression. S. Srivastava et al. [26] developed an extensible planner-independent interface layer to combine the off-the-shelf task planners and motion planners in order to facilitate the integrated planning.

The emphasis of these aforementioned methods was mainly laid on the seamless and effective integration between task planning and motion planning. They are capable of solving complicated task-motion planning problems and exhibit desirable flexibility and generality. However, when these methods are applied to robots with anthropomorphic arms, the generated movements are likely to be unpredictable and unfriendly for humans working or collaborating with these robots. For these anthropomorphic arms working in the vicinity of humans, it is suggested in this paper that anthropomorphism is incorporated as a constraint in the solution of the task-motion planning by introducing inspiring human arm motion principles, which distinguishes our proposed method from other integrated task-motion planning methods. Therefore, this paper is aimed at developing a dedicated anthropomorphism-aware task-motion planning framework for anthropomorphic arms.

To make a robot perform the outcome of task planning, i.e., a sequence of actions, there are usually infinite number of possible solutions of its joint trajectories, which is analogous to the redundancy resolution problem in the motion planning of redundant robots mentioned before. To solve this larger scale redundancy resolution problem in the integrated task-motion planning, dimensionality reduction is the key to the problem. Being aware of the fact that movement primitives have been evidenced in human arm motions by a number of neurophysiological experiments [27, 28, 29], the concept of movement

primitive is employed in our framework as an anthropomorphic and modular idea to reduce the dimentionality of the redundancy resolution problem in the task-to-joint-trajectory decomposition.

The discovery of movement primitives from neurophysiology has boosted the relevant research in the fields of computer science and robotics. A. J. Ijspeert et al. proposed an important tool termed *Dynamic Movement Primitive* (DMP) to express the observed or demonstrated motions [30, 31]. DMP is based on a general nonlinear dynamic system which possesses powerful capability of describing various joint trajectories. A learned DMP (calibrated by a reference motion) can be used to adapt to novel situations while keeping the similar shape of the reference trajectory by just tuning some simple parameters, such as the goal position and duration. This DMP framework was recently extended and applied to human-robot interaction scenario. Interaction primitives were proposed to help robots learn skills to interact with human partners in various patterns [32, 33]. Alternatively, other different techniques can be employed to express movement primitives in observed human body motions: *Principal Component Analysis* (PCA) was used to recognize and analyze temporally parallel primitives, which were also regarded as kinematic synergies [34], while *Hidden Markov Model* (HMM) was utilized to segment arm movements into temporally serial primitives [35]. The primitives obtained through aforementioned methods can be subsequently utilized to synthesize and construct whole-body motions of virtual human figures or real robots according to designed criteria and rules: Y. Li et al. proposed a motion texture method to synthesize the motion of a virtual character [36]. C. Rose et al. employed the notions of verbs and adverbs to interpolate several edited sample motions for facilitating the generation of novel motions [37]. G. Guerra et al. built a complete human activity language, which consists of kinetology, morphology and syntax, to combine recognized human movement primitives and reconstruct the same motion on a simulated human figure [38].

However, the human body motion was usually only treated as a high-dimensional data flow for the identification and segmentation of movement primitives in these methods mentioned above, and few geometrically structural features and/or anthropomorphic motion rules of human body were taken into account, which resulted in the lack of biological bases in the segmented movement primitives and corresponding organization rules. Besides, a human-involved demonstration is often required prior to the acquisition of movement primitives in this type of methods. Different from the way the movement primitives are obtained through demonstration or imitation learning, in this paper, the movement primitives and corresponding motion grammar of anthropomorphic arms are inspired by and directly designed according to several important hypotheses/rules of human arm motion in neurophysiology. A complete motion language (A^2ML) containing the movement primitives and motion grammar is built towards the automatic human-like execution of various tasks without any demonstrations. To our best knowledge, this is the first time that a human-inspired motion language is applied to the task-motion planning problem of anthropomorphic arms, which can make significant contribution to the manipulation performance enhancement of service robots with anthropomorphic arms working with humans.

This work can be referred to as the latest extension to the authors' previous work [39]. The major improvements and contribution of this work are listed as follows:

1) The previous three-level task-motion framework is extended to a four-level framework with one added movement segment level, which forms a more complete motion language.

2) Two arm configuration representation spaces: Cartesian-space-swivel-angle space and human arm triangle space are formally correlated for the movement primitive design. The mapping between the two spaces are also established. The interweavement between the designed movement primitives in the two representation spaces characterizes the proposed task-motion planning framework.

3) Rigorous motion grammar rules are proposed and expressed with a formal notation, i.e., EBNF [40].

4) Decomposition techniques from tasks to movement primitives are proposed and described by the motion grammar.

5) The new framework is equipped with one QP-based and one sampling-based methods as two powerful solvers for transforming the motion language to the joint trajectories.

6) Extensive simulations and experiments are implemented on two different anthropomorphic arms to verify and hightlight the advantages and generality of the proposed A^2ML.

3. Hypotheses/rules of human arm motion

Recently, the existence of movement primitives in human arm motion at different levels had been manifested and evidenced in many investigations in the neurophysiological field. Based on this significant finding, several hypothesis or rules are selected and their corresponding correlations with previous work in robotics and implications for this work are presented:

Rule 1*: Human arm motion consists of movement primitives* [27, 41].

The primitives are found and embodied in the human arm motion at different levels including the kinematic, dynamic, muscle, and neural levels. In this paper, the term "movement primitive" is discussed at the kinematic level.

The joint trajectories of robot motion are usually high-dimenisional data flow. The modularization is a commonly used dimensionality reduction approach to the robot task-motion planning. If an action in the task planning can be thought of as a larger motion module, a movement primitive with coordinated joint motions in the motion planning can be considered as a smaller motion module. Most of the applications of movement primitive concept to robotics belong to imitation learning where the movement ptimitives are extracted from observed human body motion data [30]. Differently, the movement primitives are directly designed from selected human arm motion hypotheses/rules in this paper. These designed primitives underpin the proposed modularized A^2ML.

Rule 2*: Human arm motion can be planned and controlled in both of the joint and Cartesian spaces* [42, 43].

A focus of attention of human arm can be maintained on the posture of the whole arm, such as the arm stretching movement,

or be directed to the position and orientation of the wrist, such as the reaching-grasping movement.

Both of the joint and Cartesian spaces are important in robotics. For an industrial robot, its path in Cartesian space is usually required for accomplishing a certain task. Its attention focus is therefore placed in Cartesian space. While for a humanoid robot with a large number of degrees of freedom, the motion planning subject to multiple hard constraints (e.g., collision avoidance and joint limits) is usually conducted in the configuration space to fully exploit its motion capability, such as sampling based methods. This rule inspires the recognition of two different but associated representation spaces of arm configuration for the movememt primitive design, and the interaction control of arm configuration in the two spaces.

Rule 3*: Human arm motion can be constrained and controlled towards a goal position/configuration or following a motion path* [44, 45].

This implies two different constraint types probably exist during the human arm motion process, i.e., goal constraint and path constraint.

These two constraint types are extensively employed in the robot motion planning. For the motion planning in Cartesian space, path constraint is common as mentioned in *Rule 2*. In the cases where a reference path is not as important as a specified goal, the problem can be reformulated as an optimization problem [46] where the tracking accuracy can be regulated by weights to allow for deviations from the reference path in the middle, but converge to the end point of the reference path in the end. For the motion planning in joint space, sampling based methods are good at searching for a solution to a goal configuration by nature. The path constraints can be added by sampling in a task-contrained configuration space [47]. This rule provides another perspective for our movement primitive design.

Rule 4*: The focus of interest of the human arm motion can be shifted during the motion process* [48, 49].

A *Movement Focus of Interest* (MFoI) is a focus of attention during a certain movement phase, which can be different in different phases. For instance, the arm rotates about the upper arm direction, and in the meantime, stretches out the hand to a specified Cartesian posture. The MFoI is shifted from the self-rotation of the upper arm to the goal hand posture.

In robotics, a series of actions decomposed from a complex task can be regarded as a MFoI shift. The MFoI is switched from one action to another. The multiple incompatible objectives with different priorities in a hierarchical quadratic optimization [13], which is used to solve the robot *Inverse Kinematics* (IK) problem, can be also considered as multiple MFoIs. The MFoI shift happens when new objectives are added or their priorities are exchanged. Inspired by this rule, the MFoI flow is employed as a significant approach to outlining the "shape" of the motion process and enriching the diversity of anthropomorphic arm motion.

Rule 5*: Motion velocity curve of human arm is characterized by a unimodal bell-shaped profile* [50, 51].

Inspired by this feature, the angular/linear velocity profiles of anthropomorphic arm, such as the angular velocity of the shoulder or the linear velocity of the wrist center, can be represented by the cosine function to simulate the unimodal bell shape. Compared to the trapezoidal velocity profile typically used for industrial robots, the human-like velocity profile would alleviate the jerky motions at the corners of the trapezoidal profile and therefore generate smoother motion.

Rule 6*: Movement primitives can be connected to each other in a sequential or parallel way* [27, 41].

These two connection patterns are natural in robotics. The connection between decomposed neighbouring actions in task planning is sequential connection, while the implementation of robot motion is realized by combining the motions of all the joints in parallel. The two natural connections are employed as fundamental ways of organizing the designed movement primitives in the proposed A^{2}ML. In addition, a mixed form of the two patterns, that is, transitional connection, is also designed for connecting movement primitives while guaranteeing the continuity of the motion.

4. Motion language for anthropomorphic arms

4.1. Overview of the framework

Inspired by the human arm motion, the six hypotheses/rules proposed in Section 3 are utilized and adapted as basic guidelines for the design and development of the task-motion planning framework for anthropomorphic arms. To fully embody these human arm motion principles and maximize the anthropomorphism, the anthropomorphic arms discussed in this paper refer to the robot arms which possess human-like shoulder-elbow-wrist configuration and have seven degrees of freedom in total. Specifically, the shoulder and wrist equivalents have three degrees of freedom, respectively, and the elbow equivalent is a single degree-of-freedom joint.

The complete framework is built as shown in Fig. 1 including five levels: task, action, movement segment, movement primitive and joint trajectory levels. The designed general A^{2}ML contains the first four levels. From top to down, a complex task, T, can be achieved by executing a sequence of actions in a logical order, which belongs to task planning. An action, A, is defined as a continuous arm motion process from a stationary state to another with a certain clear purpose, such as pushing, raising hand, etc. Subsequently, an action consists of several movment segments, which are connected in a transitional way. A movement segment, MS, is defined as a complete piece of arm movement, in which all the joint motions are specified appropriately. A movement segment is further comprised of several movement primitives in a parallel manner. A movement primitive, MP, is the fundamental element of A^{2}ML, which is inspired by *Rule 1*. A movement primitive can be the motion of a single joint of anthropomorphic arm or coordinated motions of multiple joints. Once a movement primitive is specified, the motions of the remaining joints are unspecified in our previous work [39], whereas they are defined as an attendant complementary movement primitive and planned appropriately in this paper. Therefore, the complementary movement primitive pair constitutes a complete movement segment, which necessitates the introduction of the new movement segment layer in

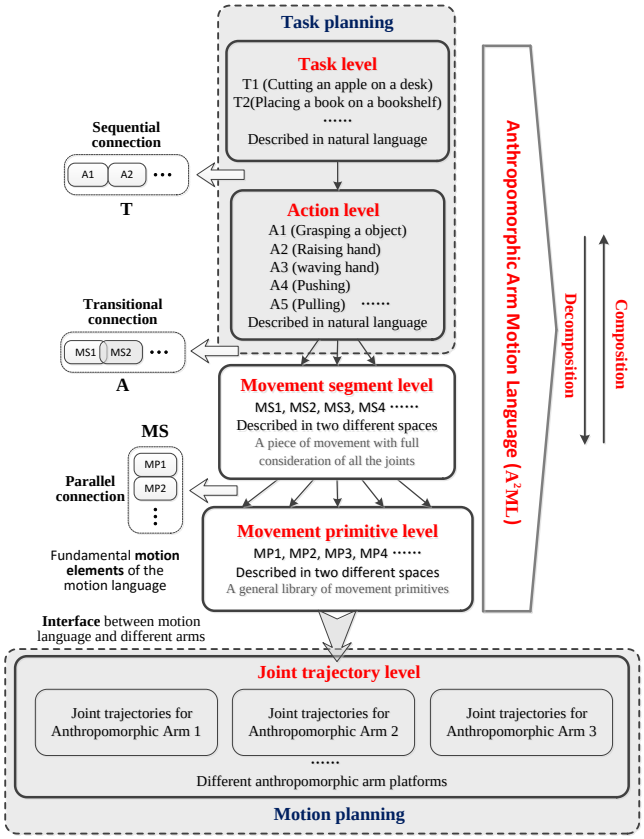

Figure 1: The proposed integrated task-motion planning framework of anthropomorphic arm and A^{2}ML.

the framework. A^{2}ML provides a general library of movement primitives which are recognized and designed in two distinct representation spaces of arm configuration. In the end, for any anthropomorphic arm with human-like physiological joints, the joint trajectories of its motion described by the movement primitives can be obtained by an interface between the language and arms, which makes the A^{2}ML available for different anthropomorphic arms. The last step refers to motion planning. It can be seen that the modular movement primitives and movement segments play essential roles in the integration between task planning and motion planning.

4.2. A^{2}ML motion elements: a library of movement primitives

A movement primitive is a motion element which reflects the continuous change of a local state of the arm configuration over a period of time. The design of movement primitive is derived from the representation of the arm posture. According to *Rule 2*, the focus of attention of the human arm would be paid on the wrist posture (position and orientation) or the configuration of the whole arm during the motion process. Inspired by this motion rule, for an anthropomorphic arm, it is apparent that the arm configuration can be determined and described from two different perspectives. One traditional way of expressing the arm configuration is to employ the wrist posture and add one supplementary variable, *Swivel Angle* [52], to complete the description of the arm posture. Swivel angle is defined as the angle between the human arm plane and the vertical plane shown

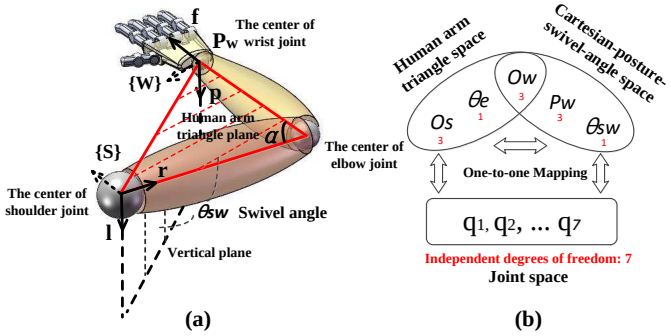

Figure 2: Anthropomorphic arm configuration represented in Human Arm Triangle (HAT) space and Cartesian-Posture-Swival-Angle (CPSA) spaces (a), and the relation between these two spaces (b).

in Fig. 2 (a). This *Cartesian-Posture-Swivel-Angle* (CPSA) representation is based on the wrist position, which is widely used in the robotics area. On the other hand, a fully position-independent expression of arm configuration termed *Human Arm Triangle* (HAT) [39] is employed to describe the arm motion in the situations where the position information is trivial, such as the waving and stretching movements. As shown in Fig. 2 (a), The human arm triangle is parameterized by five geometrical elements, i.e., $\boldsymbol{r}$, $\boldsymbol{l}$, α, $\boldsymbol{f}$ and $\boldsymbol{p}$. $\boldsymbol{r}$ denotes the unit direction vector of the upper arm, $\boldsymbol{l}$ denotes the unit normal vector of the plane of human arm triangle. The direction of $\boldsymbol{l}$ is defined by the right-hand rule, where the right-hand screw direction is the direction of elbow extension. α refers to the angle between the upper arm and forearm. $\boldsymbol{f}$ means the unit direction vector of the finger. $\boldsymbol{p}$ expresses the unit normal vector of the palm plane, the direction points outward. These elements contain thirteen scalar variables in total, since $\boldsymbol{r}$, $\boldsymbol{l}$, $\boldsymbol{f}$, and $\boldsymbol{p}$ are 3-dimensional vectors. However, one constraint in each vector is needed to meet the normalization requirement, and $\boldsymbol{r}$, $\boldsymbol{l}$ and $\boldsymbol{f}$, $\boldsymbol{p}$ are two groups of mutually perpendicular unit vectors, so there are only seven independent scalar variables left due to the six constraints in total. Hence, a HAT space spanned by these five geometrical elements has a one-to-one mapping relationship with the joint space of a 7-DoF anthropomorphic arm. However, the configuration expression in the HAT space is more intuitive than that in the joint space. It is worth noting that the five geometrical elements always exist in most of anthropomorphic arms, which enables a favorable generality and makes the expression independent of the joint configuration (the assignment and sequence of mechanical joints) and the dimension (the lengths of the upper arm and forearm) of the arm.

To further reveal the differences and correlations between the two arm configuration representations in the HAT and CPSA spaces, two frames, $\{S\}$ and $\{W\}$, are created and attached to the shoulder and the wrist shown in Fig. 2 (a). The $x-$ and $y-$ axes of frame $\{S\}$ are defined by the unit vectors $\boldsymbol{r}$ and $\boldsymbol{l}$ respectively and $\boldsymbol{f}$ and $\boldsymbol{p}$ are the $x-$ and $y-$ axes of $\{W\}$, which implies the orientations of the two frames are regulated by $\boldsymbol{r}$ and $\boldsymbol{l}$, and $\boldsymbol{f}$ and $\boldsymbol{p}$. Let symbols $\boldsymbol{O}_s$, θ_e, $\boldsymbol{O}_w$, $\boldsymbol{P}_w$ and θ_{sw} denote the orientation of the shoulder frame, the angle of human arm triangle (elbow), the orientation of the wrist frame, the position

of the wrist and the swivel angle respectively. As presented in Fig. 2 (b), it is easy to find that $\boldsymbol{O}_w$ is the common part of the two expressions and the human arm triangle is characterized by $\boldsymbol{O}_s$ and θ_e while the traditional expression is distinguished by $\boldsymbol{P}_w$ and θ_{sw}. One-to-one mapping relationship exsits between the HAT and CPSA spaces since both of them have one-to-one mapping with the original joint space.

All these five variables ($\boldsymbol{O}_s, \theta_e, \boldsymbol{O}_w, \boldsymbol{P}_w$ and θ_{sw}) can be treated as different and fundamental facets to comprehensively describe all the local states of the arm configuration in an intuitive way. Actually, in this way, seven internal 1-DoF mechanical joints without unified designation in various arms are regrouped by the two sets of variables with tangible geometrical meanings, which embodies the spirit of movement primitive. Therefore, our movement primtive design will rely on these variables and five basic movement primitives are defined as the continuous changes of these local state variables over a period of time:

$$MPB(t) = \boldsymbol{O}_s(t), \theta_e(t), \boldsymbol{O}_w(t), \boldsymbol{P}_w(t), \text{ or } \theta_{sw}(t). \tag{1}$$

Since each one of them corresponds to one part of the whole set of descriptive variables, some of them can be combined in parallel (*Rule 6*) to yield the movement primitives with multiple elementary state variables:

$$\begin{gathered}\boldsymbol{O}_s\theta_e(t),\ \boldsymbol{O}_s\boldsymbol{O}_w(t),\ \theta_e\boldsymbol{O}_w(t),\ \boldsymbol{O}_s\theta_e\boldsymbol{O}_w(t),\\ \boldsymbol{O}_w\boldsymbol{P}_w(t),\ \boldsymbol{O}_w\theta_{sw}(t),\ \boldsymbol{P}_w\theta_{sw}(t),\ \boldsymbol{O}_w\boldsymbol{P}_w\theta_{sw}(t).\end{gathered} \tag{2}$$

To avoid the conflict of description, it is defined that only the movement primitives from the same representation space can be combined in parallel. Therefore, along with the eight combined primitives, thirteen primitives are designed until now. Furthermore, inspired by *Rule 3*, movement primitives can be also classified into two categories from the perspective of constriant type: goal-directed primitives and path-constrained primitives. In the former category, only the goal state of the primtive is required and the path to the goal is not specified. On the contrary, the path of the primitive is characterized by specified features, which remain unchanged over the whole motion process in the latter category. The two categories of primitives are defined mathematically based on the basic local state variables:

$$MPB^g(t, G, T),\ MPB^p(t, P, S, T), \tag{3}$$

where G denotes the goal state of the basic goal-directed movement primitive, MPB^g. T indicates the temporal size of movement primitive, i.e., total running time. P means a constant set of the unchanged features of the basic path-constrained primitive, MPB^p. S is the adjustable spatial size of the path. For example, for a primitive $\boldsymbol{P}_w^p$ characterized by a straight line path, P is $\{'linear', \boldsymbol{d}\}$ where $\boldsymbol{d}$ is a unit vector indicating the path direction, and S is the travel distance along this direction.

As a consequence, the number of the basic movement primitives will be doubled to be ten, and the total number of movement primitives in the library including the combined ones in (2) is up to fifty[1]. And it is worth noting that all the designed movement primitives are applicable to different anthropomorphic arms with human-like physiological joints thanks to the general local state variables. In addition, since the two sets of state variables are sufficient and complete to describe the arm configuration in two different but associated representation spaces, and the two complementary constraint types fully characterize the time histories of these state variables, the proposed library of movement primitives is therefore theoretically considered to be complete.

4.3. A^{2}ML motion grammar: organization rules for anthropomorphic arm motion

After the library of movement primitives is completely well defined, the corresponding motion grammar is supposed to be defined accordingly to formally formulate how these primitives are connected and organized into a complicated task. For this purpose, *Extended Backus-Naur Form* (EBNF) [40] is selected as the notation to describe the motion grammar and motion language in a formal mathematical way. The whole motion grammar expressed in EBNF is presented in (4).

$$\begin{array}{rrcl}
1. & \boldsymbol{O}_w & := & '\boldsymbol{O}_w^{g\prime} \mid '\boldsymbol{O}_w^{p\prime};\\
2. & \boldsymbol{P}_w & := & '\boldsymbol{P}_w^{g\prime} \mid '\boldsymbol{P}_w^{p\prime};\\
3. & \theta_{sw} & := & '\theta_{sw}^{g\prime} \mid '\theta_{sw}^{p\prime};\\
4. & \boldsymbol{O}_s & := & '\boldsymbol{O}_s^{g\prime} \mid '\boldsymbol{O}_s^{p\prime};\\
5. & \theta_e & := & '\theta_e^{g\prime} \mid '\theta_e^{p\prime};\\
6. & MPB^C & := & \boldsymbol{O}_w \mid \boldsymbol{P}_w \mid \theta_{sw};\\
7. & MPB^H & := & \boldsymbol{O}_s \mid \theta_e \mid \boldsymbol{O}_w;\\
8. & MPB & := & MPB^C \mid MPB^H;\\
9. & MP^C & := & MPB^C,\ \{'+Pa',\ MPB^C\};\\
10. & MP^H & := & MPB^H,\ \{'+Pa',\ MPB^H\};\\
11. & MP & := & MP^C \mid MP^H;\\
12. & \boldsymbol{O}_w\boldsymbol{P}_w\theta_{sw} & := & \boldsymbol{O}_w,\ '+Pa',\ \boldsymbol{P}_w,\ '+Pa',\ \theta_{sw};\\
13. & \boldsymbol{O}_s\theta_e\boldsymbol{O}_w & := & \boldsymbol{O}_s,\ '+Pa',\ \theta_e,\ '+Pa',\ \boldsymbol{O}_w;\\
14. & MP^- & := & (\boldsymbol{O}_w\boldsymbol{P}_w\theta_{sw},\ '-Pa',\ MP^C)\\
15. & & & \mid (\boldsymbol{O}_s\theta_e\boldsymbol{O}_w,\ '-Pa',\ MP^H);\\
16. & MS & := & MP,\ '+Pa',\ MP^-;\\
17. & A & := & MS,\ \{'+Tr',\ MS\};\\
18. & T & := & A,\ \{'+Se',\ A\}.
\end{array} \tag{4}$$

Everything inside the single quotation signs, $' '$, is a terminal, which is the smallest inseparable element in the EBNF. Apart from the ten basic primitives, $'MPB^{g\prime}$ or $'MPB^{p\prime}$, the other terminals with signs ahead, $'+/-Pa'$, $'+Tr'$, $'+Se'$, serve as parallel, transitional and sequential connection manners in the A^{2}ML, and signs $+$ and $-$ mean the operations of adding and removing a primitive, segment or action. A nonterminal (without $' '$) can be defined ($:=$) by concatenating multiple terminals and/or nonterminals, and the concatenation operation is described by comma.

In lines 1 through 8 in (4), a basic movement primitive, MPB, is defined as either a goal-directed (MPB^g) or a ($|$) path-constrained (MPB^p) primitive with one of the five elementary state variables from either the CPSA space (MPB^C) or the HAT space (MPB^H) as mentioned in the preceding subsection. Based on this, it is explained in lines from 9 to 11 that a general movement primitive, MP, can be a single basic movement primitive

[1]The quantity can be calculated in a combination sense: $5\times C_2^1 + 6\times C_2^1\times C_2^1 + 2\times C_2^1\times C_2^1\times C_2^1$.

or consists of several basic movement primitives of different state variables by combining them in a parallel way in which these component primitives begin to be executed simultaneously and last for the same duration (curly braces {} means everything inside them can be repeated any number of times, including no repetition at all). It is worth noting that the component primitives must be from the same representation space.

In line 12 to line 15, $\boldsymbol{O}_w\boldsymbol{P}_w\theta_{sw}$ and $\boldsymbol{O}_s\theta_e\boldsymbol{O}_w$ are defined as two complete movement primitives in the CPSA and HAT spaces respectively, which are able to uniquely determine the motion process of the whole arm. A complementary movement primitive, MP^-, is defined as a primitive which will complement the corresponding original movement primitive, MP, to form $\boldsymbol{O}_w\boldsymbol{P}_w\theta_{sw}$ or $\boldsymbol{O}_s\theta_e\boldsymbol{O}_w$. Subsequently, a movement segment, MS, is defined as a complete piece of arm movement by combining MP and MP^- in a parallel way. An action, A, is defined by connecting several movement segments. Transitional connections are employed to guarantee the continuity of action. Specifically, for the transitional connection, the following movement segment would begin to be performed before the preceding one is finished. Please refers to Appendix A for its mathematical expression. In the end, a task, T, is defined by executing a sequence of actions in an appropriate order. The neighbouring actions are connected in a sequential way in which the next action would not begin to be performed until the preceding one is completely finished since the actions are relatively independent and usually have logical relations between each other. The usage of the proposed motion grammar is exemplified in Sections 5.2 and 6.3.

5. Decomposition and implementation of task

In Section 4, the elements of the A^{2}ML framework and the related motion grammar rules for organizing all these elements are defined and introduced in a bottom-up manner. However, to implement the proposed A^{2}ML framework on a real arm platform, a top-down methodology for decomposing a specified task into specific joint trajectories has to be proposed, which is discussed in this section.

5.1. Decompoisiton of a task into actions

The first step commences with the decomposition of task into actions, which falls into the task or symbolic planning domain. Therefore, it can be defined as a standard symbolic planning problem, $\mathbb{P}$, based on a STRIPS-like description [22]:

$$\mathbb{P} = (\sum, s_0, g),$$

where $\sum$ is a state transition system $\sum = (\mathbb{S}, \Lambda, \gamma)$, in which $\mathbb{S}$ is a complete set of states, and each state inside is a conjunction of propositions. Accordingly, $s_0 \in \mathbb{S}$ is an initial state in the planning problem, and g is a conjunction of propositions which defines the set of goal states $\mathbb{S}_g = \{s_g \in \mathbb{S} \mid s_g \text{ satisfies } g\}$. Λ is a set of actions where each action $A \in \Lambda$ is a group of five elements $\{name(A), precond(A), effects(A), goalcstr(A), pathcstr(A)\}$, where $name(A)$ is a character string used for naming the action, A; $precond(A)$ is a conjunction of positive propositions $precond^+(A)$ and negative propositions $precond^-(A)$, which have to be checked as the preconditions before executing an action. $effects(A)$ is a conjunction of positive propositions $effects^+(A)$ and negative propositions $effects^-(A)$ that should be added or deleted after action has been applied. $goalcstr(A)$ and $pathcstr(A)$ ($GC(A)$ and $PC(A)$ will be used as abbreviations of them in the following expression for brevity) are the geometrical goal and path constraints of the action in contrast to the topological constraints reflected in $precond(A)$ and $effects(A)$. These two elements are newly added action properties compared to the standard form, and they act as an interface to connect this decomposition step to the next decomposition step which is explained detailedly in Section 5.2. Finally, the transition function γ is built as follows. An action A can be applied on a state $s \in \mathbb{S}$ and a new state $s' \in \mathbb{S}$ can be derived if and only if:

$$\begin{aligned}
&precond^+(A) \subset s;\\
&precond^-(A) \bigcap s = \emptyset;\\
&s' = \gamma(s, A);\\
&s' = (s - effects^-(A)) \bigcup effects^+(A).
\end{aligned}$$

Once the decomposition of a task into actions is formulated in this way, classical task planners can be applied easily to get the solution, which is a sequence of actions leading the robot to move from s_0 to s_g. However, this part is not our emphasis in this paper while more attention will be paid to the remaining decomposition steps.

5.2. Decompositon of an action into movement segments and movement primitives

Inspired by *Rules 2-4*, the interaction between goal states described in the two different representation spaces and the interplay between path constraint and goal constraint expressed in the different spaces are allowed in our action design. Seven action types are then proposed and designed for anthropomorphic arms, which are summarized in Fig. 3. Each action type corresponds to different rule of decomposing action into component movement segments and movement primitives. The motion grammar introduced in Section 4.3 is used to describe all the decomposition rules. Each action type is characterized and identified by its unique geometrical constraint properties, $GC(A)$ and $PC(A)$, given by the outcome from the preceding decompositon step introduced in Section 5.1. For instance, the two properties of the action type of hierachical goal constraints (type III) can be described: $GC(A) = \{G_1 /\!/ G_2\}$ and $PC(A) = \{'void'\}$, which means the actions of this type contain two prioritized goal constraints (G_1 over G_2) but without any path constraints. All the characteristic properties of different action types are highlighted in red in Fig. 3. Therefore, the decomposition of actions into movement segments and movement primitives functions in a table lookup manner. It receives the action properties, $GC(A)$ and $PC(A)$, as inputs and returns the corresponding rule as output to decompose the action, A. The specifics of all the action types and their corresponding decomposition rules are explained in the following.

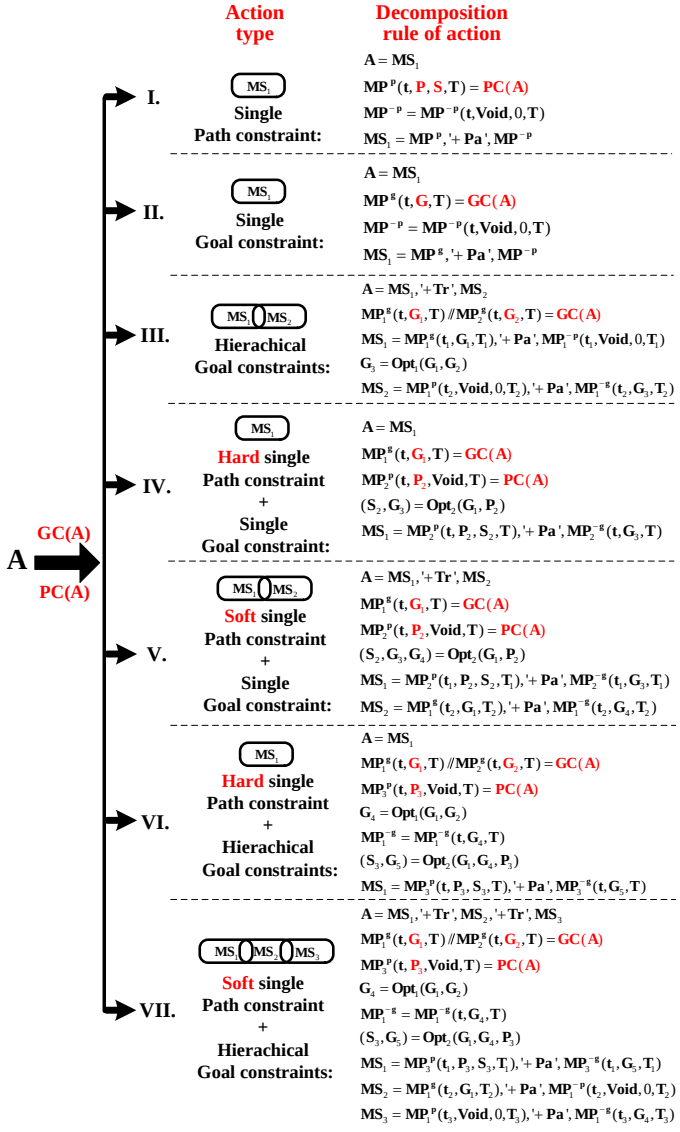

Figure 3: Schematic diagram of action decomposition rules.

Action type I only has one single path constraint[2] with $PC(A) = \{P, S\}$ and $GC(A) = \{'void'\}$. The complementary movement primitive of the featured path-constrained movement primitive ($MP^P(t, P, S, T)$) is also of path-constrained type. It keeps the remaining local state variables unchanged during the whole motion process, i.e., $MP^{-P}(t, void, 0, T)$. The featured movement primitive and its complementary movement primitive will be connected in a parallel manner to constitute a movement segment for this type of actions: $A = MS_1, MS_1 = MP^P, '+Pa', MP^{-P}$.

Actions of type II move the arm toward a single goal without any path constraints, i.e., $GC(A) = \{G\}$ and $PC(A) = \{'void'\}$. The featured movement primitive is then $MP^g(t, G, T)$, and its complementary movement primitive will keep the rest of the local state variables invariant as we deal with for the actions of type I. Parallel connection is also employed to combine the two movement primitives into a complete movement segment: $A = MS_1, MS_1 = MP^g, '+Pa', MP^{-P}$.

Action type III is proposed to deal with the situation where two conflicting goals described in the two different representation spaces (CPSA and HAT spaces) are expected to be satisfied in a hierachical way: $GC(A) = \{G_1 /\!/ G_2\}$ and $PC(A) = \{'void'\}$. In this case, higher priority will be given to G_1 over ($/\!/$) G_2. Two movement segments are designed and connected in a transitional way to form the action. The first movement segment MS_1 is made up of the movement primitive $MP_1^g(t_1, G_1, T_1)$ and its complementary primitive $MP_1^{-P}(t_1, void, 0, T_1)$. In the second movement segment MS_2, the local state variables of MP_1 will keep unchanged and the rest of the variables in the same representation space will be commanded to move to the goal state G_3, which is optimized by a solver receiving G_1 and G_2 as two inputs, $Opt_1(G_1, G_2)$. The objective of the solver is to find out the best goal state(s) of the remaining state variable(s), G_3, in addition to the constant state variables of G_1, to minimize the error between the resultant and expected values of the local state variable(s) of G_2. For instance, if $GC(A) = \{\theta_e \boldsymbol{O}_w{}^g /\!/ \boldsymbol{P}_w^g\}$, then we have $G_3 = \boldsymbol{O}_s^*$:

$$\boldsymbol{O}_s^* = \underset{\boldsymbol{O}_s}{\operatorname{argmin}} \, \| f_{\boldsymbol{P}_w}(\boldsymbol{O}_s, \theta_e \boldsymbol{O}_w{}^g) - \boldsymbol{P}_w^g \|, \tag{5}$$

where $f_{\boldsymbol{P}_w}(\boldsymbol{O}_s, \theta_e \boldsymbol{O}_w{}^g)$ means the function calculating the resultant value of $\boldsymbol{P}_w$ in the arm posture $(\boldsymbol{O}_s, \theta_e \boldsymbol{O}_w{}^g)$.

The following two action types are proposed to manage the situations where one single path constraint and one single goal constraint described in different represetatoin spaces are required in one action at the same time[3]. In action type IV, we have: $PC(A) = \{P_2, 'hard'\}$ and $GC(A) = \{G_1\}$. The indicator $'hard'$ means the path constraints can not be violated during the whole motion process. To resolve the conflict between the path constraint and the goal constraint, another solver, $Opt_2(G_1, P_2)$, is employed to optimize the length of the path, S_2, and the goal state(s) of the remaining local state variable(s), G_3, to minimize the error between the resultant and expected values of the local state variable(s) of G_1. For instance, if $GC(A) = \{\boldsymbol{O}_s \boldsymbol{O}_w{}^g\}$ and $PC(A) = \{\boldsymbol{P}_w^p\}$, we can compute the optimized $S_2 = S^*_{\boldsymbol{P}_w}$ and $G_3 = \boldsymbol{O}_w^* \theta_{sw}^*$ as follows:

$$(S^*_{\boldsymbol{P}_w}, \boldsymbol{O}_w^* \theta_{sw}^*) = \underset{S_{\boldsymbol{P}_w}, \boldsymbol{O}_w \theta_{sw}}{\operatorname{argmin}} \, \| f_{\boldsymbol{O}_s \boldsymbol{O}_w}(\boldsymbol{P}_w^p, S_{\boldsymbol{P}_w}, \boldsymbol{O}_w \theta_{sw}) - \boldsymbol{O}_s \boldsymbol{O}_w{}^g \| \tag{6}$$

It is worth noting that some hard constraints, such as joint limit and internal/external collision avoidance constraints [53, 54], have to be incorporated in the two solvers, Opt_1 and Opt_2. In fact, the two optimization solvers play important roles in the interplay between the local state variables from different

[2] Path constraint described by multiple local state variables in the same representation space is considered as a single path constraint, for instance, $\boldsymbol{P}_w \boldsymbol{O}_w{}^p$ or $\boldsymbol{O}_s \theta_e{}^p$. Therefore, we can have at most two path or goal constraints in one action under this definition. However, since single path constraint is already a relatively strong constraint, parallel or hierachical path constraints in two spaces are too heavy to be considered as an action type.

[3] Here, the path constraint and the goal constraint are supposed to be from different spaces since there will be no conflict if they are from the same space. In the latter case, the action can be completed simply by keeping the rest of the local state variable(s) constant, if any.

representation spaces, which is a distinctive strength of our proposed A^{2}ML framework. In the end, an action of this type only has one movement segment which is comprised of two movement primitives in parallel, i.e., $A = MS_1, MS_1 = MP_2^p(t, P_2, S_2, T), ' + Pa', MP_2^{-g}(t, G_3, T)$.

On the contrary, the path constraint in the action type V is considered $'soft'$ against the single goal constraint: $PC(A) = \{P_2, 'soft'\}$ and $GC(A) = \{G_1\}$. This feature means that the path constraint will be respected as much as possible but it can be violated at the end of the motion to finally satisfy the specified goal constraint. The action consists of two movement segments. The first one is the same as a movement segment in the action type IV. By a transitional connection, the second movement segment will finally lead the arm to the goal state(s): $MS_2 = MP_1^g(t_2, G_1, T_2), ' + Pa', MP_1^{-g}(t_2, G_4, T_2)$. Through the solver Opt_2, the goal value(s) of the complementary state variable(s), G_4, can be obtained by minimizing the arm configuration error relative to the goal arm configuration of MS_1, which is determined by P_2, S_2, and G_3.

Regarding the last two action types, action type VI is a combination of the action types III and IV, while action type VII combines the patterns of the action types III and V. Consequently, until now, any action can be decomposed into several movement segments with explicit description of their component movement primitives by applying the decomposition rules presented in Fig. 3.

5.3. Solving for the joint trajectories of a movement segment

After the decomposition of action into movement segments, one movement segment could have three patterns, $MS = MP^p, ' + Pa', MP^{-p}$, $MS = MP^g, ' + Pa', MP^{-g}$, and $MS = MP^p, ' + Pa', MP^{-g}$. In the first pattern, the two complementary movement primitives have to follow some path constraints, while only goal constraints are required in the second pattern. In the last pattern, it is actually a mixture of the first two patterns. One movement primitive is subject to a path constraint and the other one has a goal constraint. To address the singularity issue and possible conflicts between the path constraints and some hard constraints, for instance, joint position/velocity constraints and internal/external collision avoidance constraints, the IK problem of the movement segments of the first pattern can be formulated as a QP-based optimization problem appropriately. By contrast, sampling based methods, which excel in searching for a goal state in the configuration space in a global sense, is therefore a good fit for solving for the joint trajectories of the movement segments of the second pattern. For the movement segments of the third type, both of the two methods can be used to solve their IK problems. The comparison between these two methods in this case is introduced in Section 6.2.

The QP and sampling based methods actually serve as the interface between the A^{2}ML and different arms to transform the motion language to the specific joint trajectories respecting the path (QP-based method) and goal (sampling-based method) constraints as musch as possible. The two methods will be explained separately in the following subsections. Please note that the A^{2}ML is an open framework, any other suitable motion planners can be also employed to solve for the joint trajectories of movement segments. For instance, by introducing the idea of stochastic sampling to the optimization framework, STOMP can be utilized to help overcome the local minima problem which can be suffered by the QP-based optimization [4]. On the other hand, the solutions of the sampling based method can be further refined by gradient-based techniques, such as CHOMP.

5.3.1. Solution based on quadratic programming optimization

In a QP-based optimization framework, the IK problems for the movement segments in the CPSA and HAT spaces can be formulated as a unified optimization problem at the joint velocity level in the following[5]:

$$\begin{aligned} &\underset{\boldsymbol{Q}'}{\operatorname{argmin}} \ \frac{1}{2}\|\boldsymbol{W}(\boldsymbol{J}\boldsymbol{Q}' - \boldsymbol{V}^t)\|^2 \\ &s.t. \quad \boldsymbol{C}_{col}\boldsymbol{Q}' \leq \boldsymbol{\Delta}'_{max} \\ &\boldsymbol{f}_{min}(\boldsymbol{Q}) \leq \boldsymbol{Q}' \leq \boldsymbol{f}_{max}(\boldsymbol{Q}), \end{aligned} \tag{7}$$

where $\boldsymbol{Q}'$ stands for a column vector of the joint velocities, $[q_1', q_2', ..., q_7']^T$. $\boldsymbol{J}$ is either a Jacobian in the CPSA space, $\boldsymbol{J}_{cs}$, or a Jacobian in the HAT space, $\boldsymbol{J}_{tr}$, which maps the joint velocity vector to the velocity vector in the CPSA space, $\boldsymbol{V}_{cs} = [\boldsymbol{P}_w'^T, \boldsymbol{O}_w'^T, \theta_{sw}']^T$, or in the HAT space, $\boldsymbol{V}_{tr} = [\boldsymbol{O}_s'^T, \theta_e', \boldsymbol{O}_w'^T]^T$, respectively. $\boldsymbol{P}_w'$ is the 3-dimensional linear velocity column vector of the center of the wrist joint, θ_{sw}' and θ_e' are scalar angular velocities of the swivel angle and the elbow joint, $\boldsymbol{O}_s'$ and $\boldsymbol{O}_w'$ are the 3-dimensional angular velocity column vectors of the shoulder and wrist joints. Accordingly, $\boldsymbol{J}_{cs}$ and $\boldsymbol{J}_{tr}$ can be defined as:

$$\begin{aligned} \boldsymbol{J}_{cs} &= \begin{pmatrix} \boldsymbol{J}_{ge} \\ \boldsymbol{J}_{sw}(\boldsymbol{Q}') \end{pmatrix}, \\ \boldsymbol{J}_{tr} &= \begin{pmatrix} \boldsymbol{J}_{ge}(4:6, 1:3) & \boldsymbol{0}_{3\times 1} & \boldsymbol{0}_{3\times 3} \\ \boldsymbol{0}_{1\times 3} & \boldsymbol{l}\cdot\boldsymbol{z}_4 & \boldsymbol{0}_{1\times 3} \\ \boldsymbol{J}_{ge}(4:6, 1:3) & \boldsymbol{J}_{ge}(4:6, 4) & \boldsymbol{J}_{ge}(4:6, 5:7) \end{pmatrix}, \end{aligned} \tag{8}$$

where $\boldsymbol{J}_{ge}$ denotes the original 6×7 geometric Jacobian of the anthropomorphic arm. $\boldsymbol{J}_{sw}(\boldsymbol{Q}')$ is the last row of $\boldsymbol{J}_{cs}$, which reflects the influence of all the joints on θ_{sw}'. Please refers to Appendix B for the detailed derivation of $\boldsymbol{J}_{sw}(\boldsymbol{Q}')$. $\boldsymbol{J}_{ge}(4:6, 1:3)$ means the submatrix of $\boldsymbol{J}_{ge}$ which spans over rows 4 to 6 and columns 1 to 3. $\boldsymbol{0}_{3\times 3}$ indicates a 3×3 zero matrix. $\boldsymbol{l}\cdot\boldsymbol{z}_4$ stands for the dot product of $\boldsymbol{l}$ (the unit normal vector of the human arm triangle plane) and $\boldsymbol{z}_4$, which is the axis of the forth joint usually for the elbow extension-flexion motion in anthropomorphic arms. The result of this dot product is supposed to be 1 or -1. Please note that both of $\boldsymbol{J}_{cs}$ and $\boldsymbol{J}_{tr}$ have dimensions of

[4] The completeness of the proposed methods can be improved by incorporating suitable motion planners in this way. Achieving some form of overall completeness in task-motion planning is a challenging problem, which can be ameliorated by motion process control or action design based on some piror knowledge about the environment. The emphasis of this paper is therefore placed in the establishment of the proposed novel motion language framework.

[5] (7) can be converted into a standard QP problem as $\operatorname{argmin} \frac{1}{2}\boldsymbol{Q}'^T\boldsymbol{H}\boldsymbol{Q}' + \boldsymbol{g}^T\boldsymbol{Q}'$ with $\boldsymbol{H} = \boldsymbol{J}^T\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{J}$ and $\boldsymbol{g} = -\boldsymbol{J}^T\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{V}^t$.

7×7 because of the complete arm posture description in the two spaces. $\boldsymbol{V}^t$ in (7) indicates the target velocity vector in one of the two spaces, $\boldsymbol{V}^t_{cs}$ or $\boldsymbol{V}^t_{tr}$, and $\boldsymbol{W}$ in (7) is a 7×7 diagonal weighting matrix for weighting how much close the derivative of each local state variable has to track the corresponding target velocity. Larger weight number means that precise velocity tracking of a particular local state is more critical than those with smaller weights.

Therefore, this optimization problem will attempt to find the best joint velocity vector $\boldsymbol{Q}^{'}$ which makes the arm track $\boldsymbol{V}^t$ at best in terms of the weighting matrix $\boldsymbol{W}$, in the meantime, comply with hard constraints including the collision avoidance constraints, $\boldsymbol{C}_{col}\boldsymbol{Q}^{'} \leq \boldsymbol{\Delta}^{'}_{max}$, and the joint angle/velocity limit constraints, $\boldsymbol{f}_{min}(\boldsymbol{Q}) \leq \boldsymbol{Q}^{'} \leq \boldsymbol{f}_{max}(\boldsymbol{Q})$. Please refers to [46, 53] for details about the implementation of the two constraints.

The QP optimization based solution is designed to work at the velocity level to calculate the optimized joint velocities and then the corresponding joint trajectories by integrating the velocities. Since only spatial path/goal planning of movement segments are conducted during the decomposition of action and the spatial parameters of movement segments, P, S, and G, are output as the outcome, temporal constraints have to be added to the geometrically parameterized movement segments to calculate appropriate target velocities $\boldsymbol{V}^t$ in (7) over time in order to apply the optimization method. According to *Rule 5*, the motion velocity curve of the basic movement primitive of movement segment can be simulated by employing the form of cosine function:

$$\begin{aligned} MPB^{'}(t) &= \frac{MPB^{'}_{max}}{2}(1-\cos(\frac{2\pi t}{T})), \\ D_{MPB} &= \frac{\|MPB^{'}_{max}\|T}{2}, \end{aligned} \tag{9}$$

where $MPB^{'}$ indicates an angular speed if $MPB = \theta_{sw}$ or θ_e, a linear velocity if $MPB = \boldsymbol{P}_w$, or an angular velocity if $MPB = \boldsymbol{O}_s$ or $\boldsymbol{O}_w$. $MPB^{'}_{max}$ means the maximum speed/velocity during the whole motion process of movment primitive. T is the total duration of primitive, which appeared also in $MP^p(t,P,S,T)$ or $MP^g(t,G,T)$. D_{MPB} is the overall travel distance of movement primitive, which can be reflected by S in the path-constrained primitive. For the goal-directed primitive, D_{MPB} refers to the minimum distance between the initial state and the specified goal state, G, in the default case. This implies the shortest distance between two points in the Cartesian space or the smallest rotation angle from one orientation to another. In this case, the reference path of the goal-directed primitive is given along the "shortest-distance" direction[6]. Once D_{MPB} is determined, together with the path direction, $MPB^{'}$ can be calculated with (9) at any moment. In the end, $\boldsymbol{V}^t_{cs}$ or $\boldsymbol{V}^t_{tr}$ in (7) consists of the set of $MPB^{'}$ of all the component movement primitives of a complete movement segment in the corresponding space. To take into state errors into account and drive the actual motion path to converge to the planned one, the velocity-based kinematic control scheme [55] can be employed.

In this optimization framework, the goal constraint is considered as a weaker constraint than the path constraint. In practice, for the movement segments of the third pattern, the weight of the component path-constrained primitive is set to be a certain constant value during the whole motion. By contrast, the weights of the component goal-directed primitive at the beginning and end of the motion are configured to be the same, nevertheless, it would be tuned to be lower than the initial/final value during the motion process to permit deviation from the reference path. Regarding the movement segments belonging to the first type in which all the component movement primitives are path-constrained ones, their weights will keep the same during the whole motion process, respectively.

5.3.2. Solution of sampling-based method

Thanks to the global random sampling characteristic, sampling-based motion planning methods predominate in searching for a path in a high-dimensional configuration space to a predefined goal state with consideration of multiple constraints. The most typical method in this category, RRT, is employed in this paper. Compared to the QP-based local optimization method, the sampling-based method is a global method and it directly works at the position level (sampling in the configuration space). Because of the randomness of the generated path, the motion process towards the target configuration is usually not controlled, which is consistent with the feature of the movement segments of the second pattern. For these movement segments, since the goal states of the component goal-directed movement primitives can be described in the two spaces, two IKs from the CPSA and HAT spaces to the configuration space are required to apply the sampling-based methods. Regarding the movement segments of the third type, an additional constraint, P, is imposed by the component path-constrained movement primitive. In this case, the arm posture can be sampled directly in a constrained CPSA or HAT space [56]. For instance, one movement segment in CPSA space consists of one path-constrained movement primitive $\boldsymbol{P}^p_w(t, P_{\boldsymbol{P}_w}, S_{\boldsymbol{P}_w}, T)$ and one goal-directed movement primitive $\boldsymbol{O}_w\theta_{sw}{}^g(t, G_{\boldsymbol{O}_w\theta_{sw}}, T)$, which is actually the movemet segment example employed in Section 6.2. For sampling the arm configurations of this movement segment, five variables, $S_{\boldsymbol{P}_w}$, θ_{sw} and Roll, Pitch, Yaw Euler angles[7] for expressing $\boldsymbol{O}_w$, can be utilized. With the path constraint $P_{\boldsymbol{P}_w}$ and the initial state of the arm, the corresponding configurations of the CPSA representation sampled in the constrained CPSA space (constrained by $P_{\boldsymbol{P}_w}$ in this case) can be obtained easily. Likewise, two IKs as mentioned above are needed to get the specific joint angles in order to check the feasibility of the sampled configuration with consideration of the joint limit, obstacle avoidance and so on.

Since the IK algorithm from the HAT space to the configuration space was already developed and investigated extensively for differrent anthropomorphic arms in our previous work

[6] Other default reference paths can be also given in advance for a specific goal-directed primitive.

[7] Sampling in Quaternion can be also employed to avoid the expression singularity of the Euler angles.

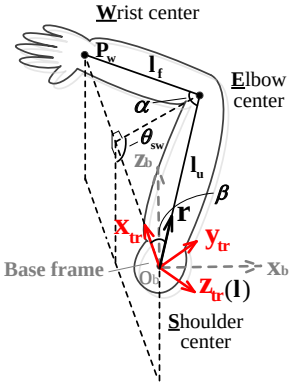

Figure 4: Schematic diagram of the mapping relationship between the CPSA and HAT spaces (excluding the orientation of the wrist).

[39, 57, 58], our strategy for deriving the IK algorithm from the CPSA space to the configuration space is to build a mapping relationship from the CPSA space to the HAT space. Because $\boldsymbol{O}_w$ is the common part in the two spaces, only the mapping relationship between $\boldsymbol{P}_w$, θ_{sw} and $\boldsymbol{r}$, $\boldsymbol{l}(\boldsymbol{O}_s)$, $\alpha(\theta_e)$ will be discussed here. As shown in Fig. 4, assume a frame $\{tr\}$ is attached to the human arm triangle with the origin at the center of the shoulder. Its x axis, $\boldsymbol{x}_{tr}$, is defined along the shoulder-wrist direction, $\boldsymbol{z}_{tr}$ is the same as $\boldsymbol{l}$. Therefore, when the swivel angle of the anthropomorphic arm is θ_{sw}, the rotation matrix of this frame can be derived:

$$\begin{aligned} \boldsymbol{x}_{tr}(0) &= \boldsymbol{P}_w/\|\boldsymbol{P}_w\|, \\ \boldsymbol{z}_{tr}(0) &= (\boldsymbol{P}_w \times (-\boldsymbol{z}_b))/\|\boldsymbol{P}_w \times (-\boldsymbol{z}_b)\|, \\ \boldsymbol{y}_{tr}(0) &= \boldsymbol{z}_{tr}(0) \times \boldsymbol{x}_{tr}(0), \\ \boldsymbol{R}_{tr}(0) &= [\boldsymbol{x}_{tr}(0),\ \boldsymbol{y}_{tr}(0),\ \boldsymbol{z}_{tr}(0)], \\ \boldsymbol{R}_{tr}(\theta_{sw}) &= \boldsymbol{R}_{tr}(0)\boldsymbol{R}_x(-\theta_{sw}), \end{aligned} \tag{10}$$

where $\boldsymbol{z}_b$ is the z axis of the base frame (vertical upward), $\boldsymbol{R}_{tr}(0)$ means the rotation matrix of frame $\{tr\}$ when the swivel angle is 0, $\boldsymbol{R}_x(-\theta_{sw})$ indicates the rotation matrix caused by the rotation about x axis by $-\theta_{sw}$ and operator $\times$ denotes the cross product. Subsequently, $\boldsymbol{r}$, $\boldsymbol{l}$ and α can be computed as:

$$\begin{aligned} \beta &= \arccos\left(\frac{l_u^2 + \|\boldsymbol{P}_w\|^2 - l_f^2}{2l_u\|\boldsymbol{P}_w\|}\right), \\ \boldsymbol{r} &= \boldsymbol{R}_{tr}(\theta_{sw})[\cos\beta,\ \sin\beta,\ 0]^T, \\ \boldsymbol{l} &= \boldsymbol{R}_{tr}(\theta_{sw})[0,\ 0,\ 1]^T, \\ \alpha &= \arccos\left(\frac{l_u^2 + l_f^2 - \|\boldsymbol{P}_w\|^2}{2l_u l_f}\right), \end{aligned} \tag{11}$$

where β is the angle between the direction of the upper arm and the shoulder-wrist direction. l_u and l_f are the lengths of the upper arm and forearm, respectively. Once the IK algorithms of the two spaces are established, the sampling-based methods can be applied conveniently to solve for the joint trajectories of the movement segments of the second and third patterns.

6. Simulations and experiments

6.1. Anthropomorphic arm platforms

The proposed A^{2}ML can be applied to different anthropomorphic arm platforms. In this section, various simulations and experiments are designed and conducted to validate the generality of the whole framework and demonstrate its desirable features and advantages. One 7-DoF virtual right arm of a quadrupedal robot, *Centauro* [59], and one 7-DoF real right arm of a humanoid robot, *Coman* [60], are employed for this purpose. As shown in Fig. 5 and Fig. 6, the two arms vary in the rotation axis designation, and have distinct rotation arrangement designs for the human-like shoulder and wrist joints (rotation axes z_1, z_2, z_3 for the shoulder and axes z_5, z_6, z_7 for the wrist). Special forward (30°) and upward (10°) tilt angles are designed for the Centauro right arm in order to enlarge its working space. In addition, they also exhibit a difference in the link dimension. However, apart from these differences, it is still able to express the postures of the two arms in both of the CPSA and HAT spaces thanks to the generality of these two spaces.

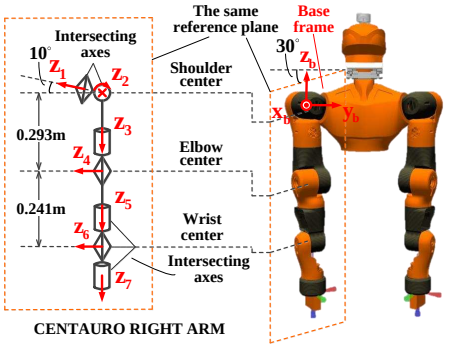

Figure 5: 7-DoF virtual Centauro right arm and the schematic diagram of its kinematic chain.

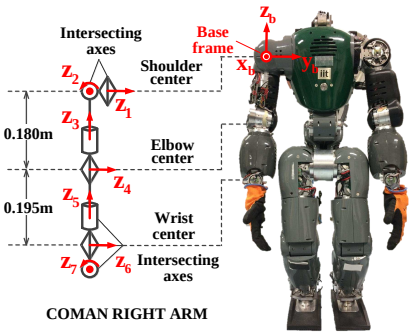

Figure 6: 7-DoF real Coman right arm and the schematic diagram of its kinematic chain.

6.2. Comparison between QP-based solution and sampling-based solution

In Section 5.3, it was mentioned that both of the QP-based optimization method and the sampling-based method can be employed to solve for the joint trajectories of the movement segment containing one path-constrained primitive and one goal-directed primitive (the third pattern). In this subsection, several simulations are designed and implemented on the virtual Centauro right arm to compare the two methods and reveal their characteristics.

The same action of type IV is employed in three different simulations. The motion of the wrist center along a specified

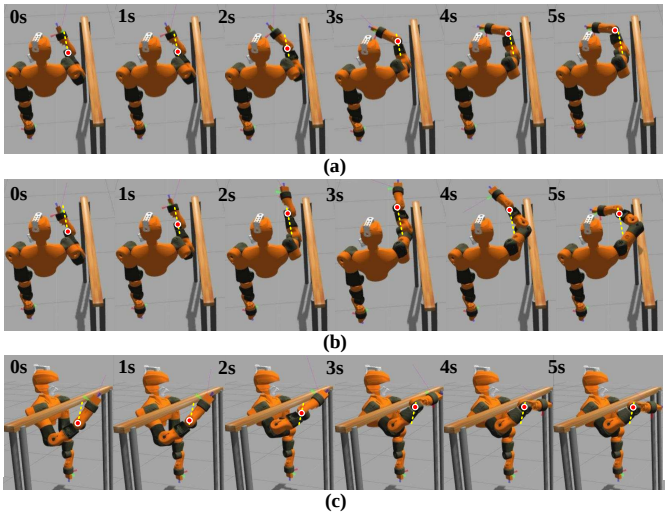

Figure 7: Snapshots of the motions of an action of type IV with the virtual Centauro right arm in three different situations. (a) The arm motion obtained by using the QP-based optimization method with a forward bar as an obstacle. (b) The arm motion obtained by the RRT-based method with a forward bar as an obstacle. (c) The arm motion obtained by the QP-based optimization method with a lateral bar as an obstacle. Dashed yellow lines stand for the desired paths and red dots indicate the current positions of the wrist center.

direction, [0.356, 0.0, 0.935], is defined as the hard path constraint of the action, and the single goal constraint is defined in the HAT space. According to the corresponding action decomposition rule, the travel distance of the wrist and the goal states of the complementary local state variables in the CPSA space (the wrist orientation and swivel angle) can be calculated by the optimization solver Opt_2. Consequently, the collision-free goal configuration of the whole arm turns out to be located above the bar as shown in the last frame in Fig. 7 (b). In the QP-based optimization method, the orientation change "path" of the wrist uses the default "shortest-distance" direction, which can be calculated by the equivalent axis-angle representation of the wrist rotation. To respect the wrist motion direction constraint as much as possible, the initial weights of $\boldsymbol{P}_w, \boldsymbol{O}_w$ and θ_{sw} are set to $10, 1$ and 1 respectively. In the sampling-based method, RRT method is employed and the sampling is executed in the CPSA space considering the same motion direction constraint of the wrist. In the first simulation, the QP-based method is employed and a forward wooden bar is set next to the Centauro right arm as an obstacle. In the second simulation, the RRT-based method is used with the same setup as the first simulation. The obstacle bar is rotated about the vertical direction by 90° and placed in front of the robot in the third simulation. In this case, since no exact solution exists, only the QP-based method can be used whereas the RRT-based method fails. In all the three simulations, the same initial and expected goal collision-free configurations of the arm are employed.

As shown in Fig. 7 and Fig. 8, the arm is able to track the specified wrist path in the early stage of the motion in the first simulation ((a) in Fig. 7). However, the arm can not reach the goal configuration and the goal position of the wrist is slightly modified and the swivel angle trajectory is heavilly restrained from the reference profile to avoid the obstacle bar. In the sec-

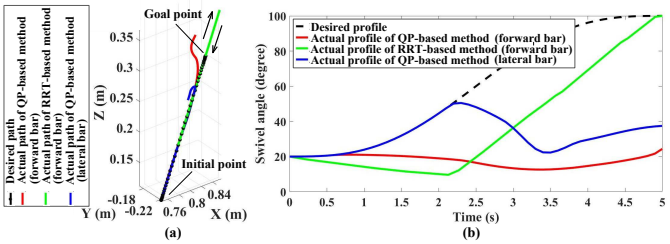

Figure 8: The corresponding paths of the wrist center (a) and the profiles of the swivel angle (b) in the three situations presented in Fig. 7. The dashed black lines stand for the desired path and profile, and the red, green and blue lines reflect the motion processes in situations (a), (b) and (c) in Fig. 7 respectively.

ond simulation ((b) in Fig. 7), thanks to the powerful global configuration searching capability, RRT-based method can lead the robot to find a solution to the goal configuration by first stretching the arm and retracting it later. In this way, the robot is able to bypass the obstacle bar and place the arm above it in the end. Note that the wrist center exceeds the goal position during the arm stretching phase but the path constraint is respected all the time. In the third simulation ((c) in Fig. 7), the wrist of the arm almost follows the specified path but stops in the middle between the initial and goal points due to the obstacle bar, and the swivel angle profile undoubtedly diverges halfway from the reference one. In contrast with the relatively precise path tracking of the wrist, the inverted bell-shaped curves of the swivel angle in the late stage of the motion in the first and third simulations reflect the attempts to track the reference profile back ((b) in Fig. 8), which benefits from the design of the weight regulation of the goal-directed movement primitive in the QP-based method.

Some distinctions between the two methods can be observed from the comparative simulations and several conclusions can be drawn as follows:

1. *Solution-searching ability*: thanks to the global sampling characteristic, RRT-based method (sampling-based method) is able to find the solution successfully if the solution exists ((b) in Fig. 7). The QP-based local optimization method, on the contrary, may fail when the connectivity of the feasible configuration space of the arm becomes undesirable and complex due to the obstacles ((a) in Fig. 7).

2. *Constraint/task modification*: Since the IK problem is considered as an optimization problem in the QP-based method, the path constraint and target configuration can be modified and an optimized closest solution can be acquired if the task is hard to achieve accurately. This feature can be nevertheless a benefit if the solution of the task does not exist ((c) in Fig. 7). On the other side, RRT-based method would fail to find the solution because the goal/path constraints have to be respected all the time without any relaxation in this kind of methods.

3. *Computation efficiency*: since RRT is a global sampling-based motion planning method, it is relatively time-consuming for the global sampling and the feasibility checks of the sampled configurations. By contrast, some

highly-efficient solvers are available to make the standard QP optimization problem solved even in real time.

Therefore, each of the two methods possesses some advantages. One can select the more suitable method according to the requirements of the specific situation.

6.3. *Comparison with a traditional method in terms of joint trajectory generation*

In this subsection, solving for the joint trajectories of Coman right arm in two different actions of type VII is first elaborated as an illustration of transforming a parameterized action to the corresponding joint trajectories. The desirable attributes of the proposed method are further highlighted in an obstacle-avoidance experiment by comparsion with the GPM in terms of the joint trajectory generation.

Through the proposed motion grammar in Fig. 3, the two actions of type VII (shown in Fig. 9 (a) and (b)) can be described in a unified form as follows, which is also visualized in Fig. 9:

$$\begin{aligned} MS_1 &= \boldsymbol{O}_s{}^{p}(t_1,P_3,S_3,T_1),'+Pa',\boldsymbol{O}_s{}^{-g}(t_1,G_5,T_1)\\ MS_2 &= \boldsymbol{P}_w\boldsymbol{O}_w{}^{g}(t_2,G_1,T_2),'+Pa',\boldsymbol{P}_w\boldsymbol{O}_w{}^{-p}(t_2,void,0,T_2)\\ MS_3 &= \boldsymbol{P}_w\boldsymbol{O}_w{}^{p}(t_3,void,0,T_3),'+Pa',\boldsymbol{P}_w\boldsymbol{O}_w{}^{-g}(t_3,G_4,T_3)\\ A_{\mathrm{VII}} &= MS_1,'+Tr',MS_2,'+Tr',MS_3. \end{aligned} \tag{12}$$

The geometrical constraint properties of both actions, i.e., known parameters, are: $GC(A_{\mathrm{VII}})=\{G_1 /\!/ G_2\}$ and $PC(A_{\mathrm{VII}})=\{P_3,'soft'\}$, and the two actions are distinguished by different path constraints, P_3, in $\boldsymbol{O}_s^p$. The calculations of all the unknown parameters in (12) are explained in detail in the following.

The two actions are started with the same configuration and targeted at the same hierachical goal constraints, in which the goal constraint with higher priority, G_1, is $\boldsymbol{P}_w\boldsymbol{O}_w$ with the wrist position, $\boldsymbol{P}_w=[0.27,-0.10,0.05](m)$, and the wrist orientation $\boldsymbol{O}_w$ described by $\boldsymbol{f}$ pointing forward and $\boldsymbol{p}$ leftward. The other goal constraint with lower priority, G_2, is defined as $\boldsymbol{O}_s$ contrarily in the HAT space. Specifically, $\boldsymbol{r}$ is expected to point in the direction of $[\cos(45^\circ),-\sin(45^\circ),0]$ and $\boldsymbol{l}$ is supposed to be downward. The goals G_1 and G_2 are expressed schematically in the last frame in Fig. 9. According to the decomposition rule of action type VII, the goal state of the complementary movement primitive of $MP_1(\boldsymbol{P}_w\boldsymbol{O}_w)$, G_4 (θ_{sw} in this case), is calculated by the optimization solver, $Opt_1(\boldsymbol{P}_w\boldsymbol{O}_w,\boldsymbol{O}_s)$, that is:

$$G_4=\underset{\theta_{sw}}{\operatorname{argmin}}\ \|f_{\boldsymbol{O}_s}(\theta_{sw},\boldsymbol{P}_w\boldsymbol{O}_w)-\boldsymbol{O}_s\|. \tag{13}$$

The optimized result of this problem is $G_4(\theta_{sw})=63.2^\circ$. Up to now, the parameters of MS_2 and MS_3 of the two actions are determined. Except for the common hierachical goals, the two actions have different path constraints P_3, which are described as the rotations of the upper arm about its own direction ($\boldsymbol{r}$ for action (a)) and the vertical direction ($-\boldsymbol{z}_b$ for action (b)), respectively, shown in Fig. 9. Subsequently, the closest configurations to the same goal configuration (G_1,G_4) subject to the path constraints mentioned above need to be found in both actions, respectively. In other words, parameters S_3 and G_5 ($S_{\boldsymbol{O}_s}$ and $\theta_e\boldsymbol{O}_w$ in this case) in MS_1 have to be calculated to complete the parameterization of actions (a) and (b) through the optimization solver, $Opt_2(G_1,G_4,P_3)(P_3=\boldsymbol{r}$ or $-\boldsymbol{z}_b)$:

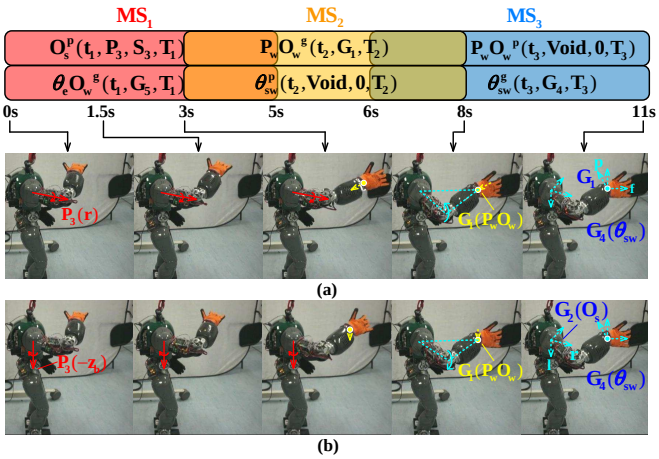

Figure 9: Snapshots of the motions of two different actions of type VII with the real Coman right arm. MS_1, MS_2 and MS_3 are three component movement segments of both of the actions. Actions (a) and (b) are characterized by the arm rotations about the upper arm and the vertical direction respectively in MS_1. The red arrow lines indicate the rotation directions of the upper arm in MS_1, the yellow lines denote the motion directions of the wrist in MS_2, and the blue arrow arcs imply the changes of the swivel angle in MS_3. The two actions have the same initial and goal arm configurations.

$$(S_3,G_5)=\underset{S_{\boldsymbol{O}_s},\theta_e\boldsymbol{O}_w}{\operatorname{argmin}}\ \|f_{\boldsymbol{P}_w\boldsymbol{O}_w\theta_{sw}}(P_3,S_{\boldsymbol{O}_s},\theta_e\boldsymbol{O}_w)-G_1G_4\| \tag{14}$$

The optimized rotation angles of the shoulder, $S_{\boldsymbol{O}_s}$, turn out to be 24.7° about $\boldsymbol{r}$ in action (a), and 37.8° about $-\boldsymbol{z}_b$ in action (b). Correspondingly, the optimized goal states of θ_{sw} and the roll-pitch-yaw Euler angles of $\boldsymbol{O}_w$ are 158.5°, -15.5°(roll), -11.5°(pitch), -104.6°(yaw) for action (a) and 102.9°, -11.8°(roll), -11.8°(pitch), -91.6°(yaw) for action (b), which means different closest configurations to the same goal configuration, G_1G_4, are generated under distinct path constraints. Via these different intermediate configurations, various motion processes from the same initial configuration to the same goal configuration can be achieved. This feature allows for the elaborate control of the arm motion process, which will highly enrich the planning and design of actions. The motion processes of the two actions are shown in Fig. 9. It is worth noting that the two optimization solvers, Opt_1 and Opt_2, play important roles in interweaving and interacting the arm configuration controls in the CPSA and HAT spaces.

To further highlight the benefits of the proposed method, another comparative experiment in a collision-avoidance scenario with the GPM, is conducted. A column obstacle with a radius of $0.03m$ is located at $[0.37,\ 0.05](m)$ on the ground with respect to the base frame of the robot. The same action (with the same parameters) as the one in Fig. 9 (a) is employed. However, different from the handling that the joint trajectories of all the movement segments are solved by the QP-based method in the previous case, the joint trajectories of MS_1 are solved by the RRT-based method in this case in order to enhance the

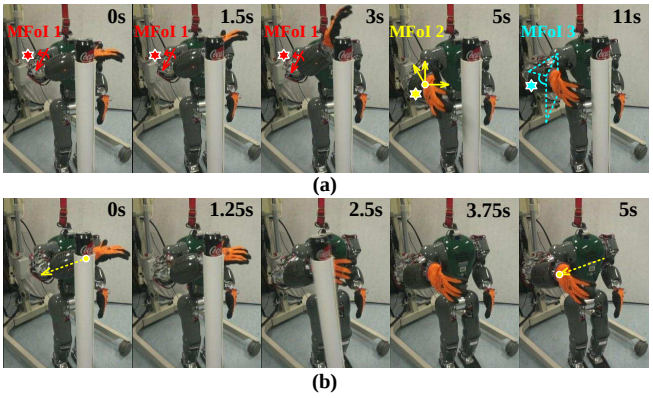

Figure 10: Snapshots of the comparative experiments on the Coman right arm in terms of obstacle avoidance ability. (a) Coman avoids the colomn obstacle successfully during the process of moving from the same initial configuration to the same goal configuration as those shown in Fig. 9. The same action as the one in Fig. 9 (a) is employed with the joint trajectories of MS_1 solved by the RRT-based method instead for better obstacle avoidance. The stars in different colors signify the shift of the movement focus of interest during the whole action. (b) Coman fails to avoid the same obstacle by using the GPM. The yellow dashed line denotes the motion path of the wrist center. The same initial arm configuration and goal hand posture are used in cases (a) and (b).

obstacle-avoidance capability of the arm. GPM is well known as a redundancy resolution approach, which is usually used to deal with the motion planning problem of a robot with one or multiple redundant DoF(s). The method works at the velocity level and can be formulated as follows:

$$\boldsymbol{Q}^{'} = \boldsymbol{J}^{+}\boldsymbol{X}^{'} + k(\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J})\nabla H(\boldsymbol{Q}), \tag{15}$$

where $\boldsymbol{J}^{+}$ means the Pseudoinverse of the Jacobian $\boldsymbol{J}$, which maps the desired velocity in Cartesian space $\boldsymbol{X}^{'}$ into a minimum-norm solution, $\boldsymbol{J}^{+}\boldsymbol{X}^{'}$. $\nabla H(\boldsymbol{Q})$ is a vector-valued function of the joint vector $\boldsymbol{Q}$, which indicates the gradient of a scalar-valued criterion function $H(\boldsymbol{Q})$. The gradient is projected into the null space of $\boldsymbol{J}$ by $(\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J})$ to form a null-space solution, i.e., $k(\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J})\nabla H(\boldsymbol{Q})$, which would maximize H as much as possbile while not affecting $\boldsymbol{X}^{'}$. k is a gain used to regulate the level of the local optimization. In this experiment, the criterion H to be optimized is defined as:

$$H(\boldsymbol{Q}) = -(\boldsymbol{Q} - \bar{\boldsymbol{Q}})^{T}(\boldsymbol{Q} - \bar{\boldsymbol{Q}}), \tag{16}$$

where $\bar{\boldsymbol{Q}}$ indicates a column vector of the median angles of all the joints. GPM with this criterion attempts to center each of the joints as much as possible while executing the specified task in Cartesian space at a higher priority. $k = 10$ is employed in the experiment. The path of the wrist is initially a line segment and the wrist orientation follows the "shortest-distance" direction.

The two methods share one initial configuration which is the same as the one employed in Fig. 9. The goal configuration is also the same as the one in Fig. 9 in the proposed method while only the same goal hand posture is set in the GPM since the swivel angle is usually not able to be precisely controlled in the GPM. As shown in Fig. 10 (a), Coman is able to avoid the obstacle successfully by using the proposed method and the corresponding minimum distance profiles of the upper arm, forearm

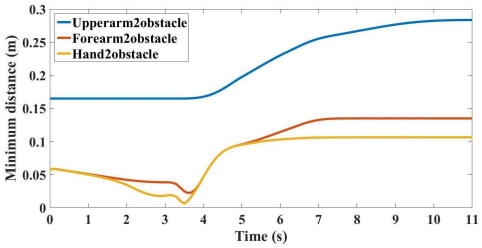

Figure 11: The minimum distance profiles of the upper arm, forearm and hand from the colomn obstacle respectively in the Coman's action in Fig. 10 (a).

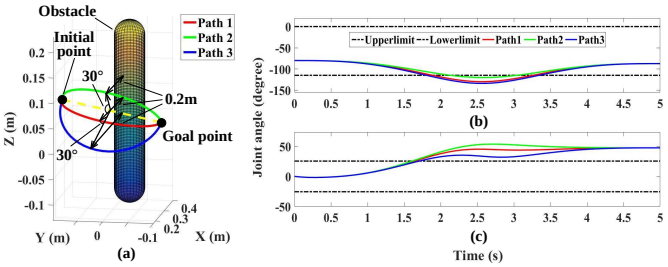

Figure 12: Three modified collision-free semi-elliptic paths of the wrist center (a) for the Coman's motion in Fig. 10 (b) and the corresponding infeasible joint trajectories of the fourth joint (the extension-flexion joint of the elbow) (b) and the sixth joint (the abduction-adduction joint of the wrist) (c).

and hand from the obstacle are shown in Fig. 11. On the contrary, as shown in Fig. 10 (b), the robot fails to avoid the same obstacle as expected by using the GPM since the column obstacle is located in the middle of the wrist path. Three modified collision-free semi-elliptic paths are subsequently tested. As shown in Fig. 12 (a), the angles between the planes of the green and red paths, and between the planes of the red and blue paths are 30°, respectively. The maximum distances of all the three paths from the central axis of the column obstacle in x direction are the same, which is $0.2m$[8]. The resultant joint trajectories of the fourth and sixth joints by the GPM for the three paths are shown in Fig. 10 (b) and (c), respectively. These infeasible joint trajectories manifest that the GPM with the local optimization of the joint angles towards the median values is not capable of guaranteeing the satisfaction of the joint limit constraint. The outcome of the comparative experiments reveals an important distinction between the two methods. In the proposed method, the action is composed of three movement segments. Their featured path and goal constraints can be considered as the MFoIs, which are mentioned in *Rule 4*. In this case, each movement segment has one different MFoI, which is $P_3(\boldsymbol{r})$, $G_1(\boldsymbol{P}_w\boldsymbol{O}_w)$ and $G_4(\theta_{sw})$ in the order of motion shown in Fig. 10 (a). This shift of the MFoI during the motion is actually a subtle balance between the motion process control and the adaptation to some hard constraints (e.g., the joint limit and obstacle avoidance in this case). To control the motion process, some key local path and goal constraints, which only occupy part of the arm DoFs,

[8] x axis perpendicular to the vertical plane determined by the initial and goal points. $0.2m$ is actually still a risky distance with consideration of the radius of the obstacle since the length of the hand is $0.2m$. However, we assume this risk can be eliminated by the further path tuning of wrist orientation.

are set to outline the "shape" of the motion. In the meantime, the remaining DoFs are relaxed to fulfil the hard constraints. Therefore, the failure of the obstacle-avoidance experiment using the GPM can be attributed to the excessive path planning of the wrist and the lack of the enough redundancy for the satisfaction of the hard constraints.

6.4. Task implementation on different arm platforms by the same motion sturcture described in A^{2}ML

In this experiment, a complete task is executed with the Centauro and Coman right arms respectively by using the proposed A^{2}ML framework in order to validate the effectiveness and generality of the framework.

As shown in the first frame of Fig. 13 (b) and (c), A table is placed in front of the robot. Under the table, there is a trash can on the ground. Box A and Box B are put on the table next to each other. It is assumed that the robot grasps something in its right hand at the beginning. The goal of the task is to push Box A aside leftwards. We assume that the same sequence of actions as shown in Fig. 13 (a) are obtained in advance by a standard task planner for the two arms according to the same topological relationships shown in the virtual and realistic environment settings. Our emphasis is nevertheless placed on how to transform this sequence of actions to the corresponding joint trajectories of the two different arms by utilizing the geometrical constraint properties of these obtained actions, that is, $GC(\mathrm{Ai})$ and $PC(\mathrm{Ai})$ $(i = 1,2,...,7)$. Each action will be explained in order as follows with reference to the motion grammar in Fig. 3:

A1: *Drop the things in the hand*: With the geometrical constraints of Action A1, $GC(\mathrm{A1}) = \{\boldsymbol{P}_w\boldsymbol{O}_w \mathbin{/\!/} \boldsymbol{O}_s\}$ and $PC(\mathrm{A1}) = \{'void'\}$, A1 can be easily recognized as an action of type III. The goal constraint with higher priority is the posture of the wrist, which is located in the middle of the table and the trash can with the palm facing the trash can. The other constraint, i.e., the orientation of the shoulder, is expected to make the upper arm close to the torso and stay away from the table as much as possible.

A2: *Retract the hand*: $GC(\mathrm{A2}) = \{\boldsymbol{P}_w\boldsymbol{O}_w\}$ and $PC(\mathrm{A2}) = \{\boldsymbol{O}_s\theta_e, 'hard'\}$. The action of type IV makes the robot rotate the shoulder about the vertical direction and keep the elbow joint static to try to reach a specified wrist posture on the right-hand side of the torso as close as possible.

A3: *Raise the hand above the table*: $GC(\mathrm{A3}) = \{\boldsymbol{P}_w\theta_{sw}\}$ and $PC(\mathrm{A3}) = \{\boldsymbol{O}_s, 'soft'\}$. The action of type V requires the robot to raise the whole arm by rotating the shoulder about $\boldsymbol{l}$ to reach a specified position above the table with the swivel angle equal to 90° as much as possible.

A4: *Approach Box B*: $GC(\mathrm{A4}) = \{\boldsymbol{P}_w\boldsymbol{O}_w\theta_{sw}\}$ and $PC(\mathrm{A4}) = \{'void'\}$. The action of type II commands the robot to move close to Box B with prepared arm posture for moving it in the following action.

A5: *Move Box B*: $GC(\mathrm{A5}) = \{'void'\}$ and $PC(\mathrm{A5}) = \{\boldsymbol{O}_s\theta_e\boldsymbol{O}_w\}$. The action of type I is employed to move Box B. The shoulder keeps static while both of the elbow and wrist joints are rotated about $\boldsymbol{l}$.

A6: *Approach Box A*: $GC(\mathrm{A6}) = \{\boldsymbol{P}_w\boldsymbol{O}_w\}$ and $PC(\mathrm{A6}) = \{\boldsymbol{O}_s\theta_e\boldsymbol{O}_w, 'soft'\}$. The action of type V reversely executes the motion of the preceding action towards a specified hand posture close to Box A for the preparation for pushing it in the final action.

A7: *Push Box A*: $GC(\mathrm{A7}) = \{'void'\}$ and $PC(\mathrm{A7}) = \{\boldsymbol{P}_w\boldsymbol{O}_w\theta_{sw}\}$. The last action of type I pushes Box A leftwards along a horizontal straight line with constant hand orientation and swivel angle.

As illustrated in Section 6.3, all the parameters of the component movement primitives of the actions above (i.e., G, P and S in (3)) can be determined by using the action decomposition rules introduced in Section 5.2. For each movement primitive, the target velocity vector $\boldsymbol{V}^t$ at any moment in (7) can be subsequently calculated based on these obtained G, P and S according to (9). The corresponding joint trajectories of each movement primitive are then solved by the QP-based method by using (7) (since the solution-searching process is not very demanding in this case, only QP-based method is used to solve for the joint trajectories). The specific joint trajectories of the Centauro and Coman arms for the whole task are shown in Fig. 13 (d) and (e), respectively. Significant differences in the joint trajectories between the two arms are observed, which are mainly attributed to the different arm kinematic chains. This arm-specific information is embodied in $\boldsymbol{J}$ in (7). It can be seen that the execution of the same task with the same motion structure on different arm platforms is feasible with consideration of complying with distinct joint limits, which proved the generality of the proposed framework. More details about all the simulations and experiments in Section 6 can be found in the attached video.

6.5. Summary

With all the results of the simulations and experiments in this Section, the features and advantages of the proposed A^{2}ML framework can be summarized as follows:

1. *Interaction between two representation spaces of arm configuration*: One of the most important features of the proposed method is that two different but associated representation spaces, i.e., the CPSA and HAT spaces, are proposed to define and describe the general movement primitives of anthropomorphic arms. With the interaction between the two spaces in terms of the arm configuration control reflected in the decomposition of action, the arm configuration control becomes rich and comprehensive.

2. *Motion process control and action design*: The arm configuration control and the interaction in the two representation spaces also enable richer motion process control of anthropomorphic arm. In conjunction with the design of the goal-directed and path-constrained movement primitives and the formalization of different action types, various motion processes from the same initial configuration

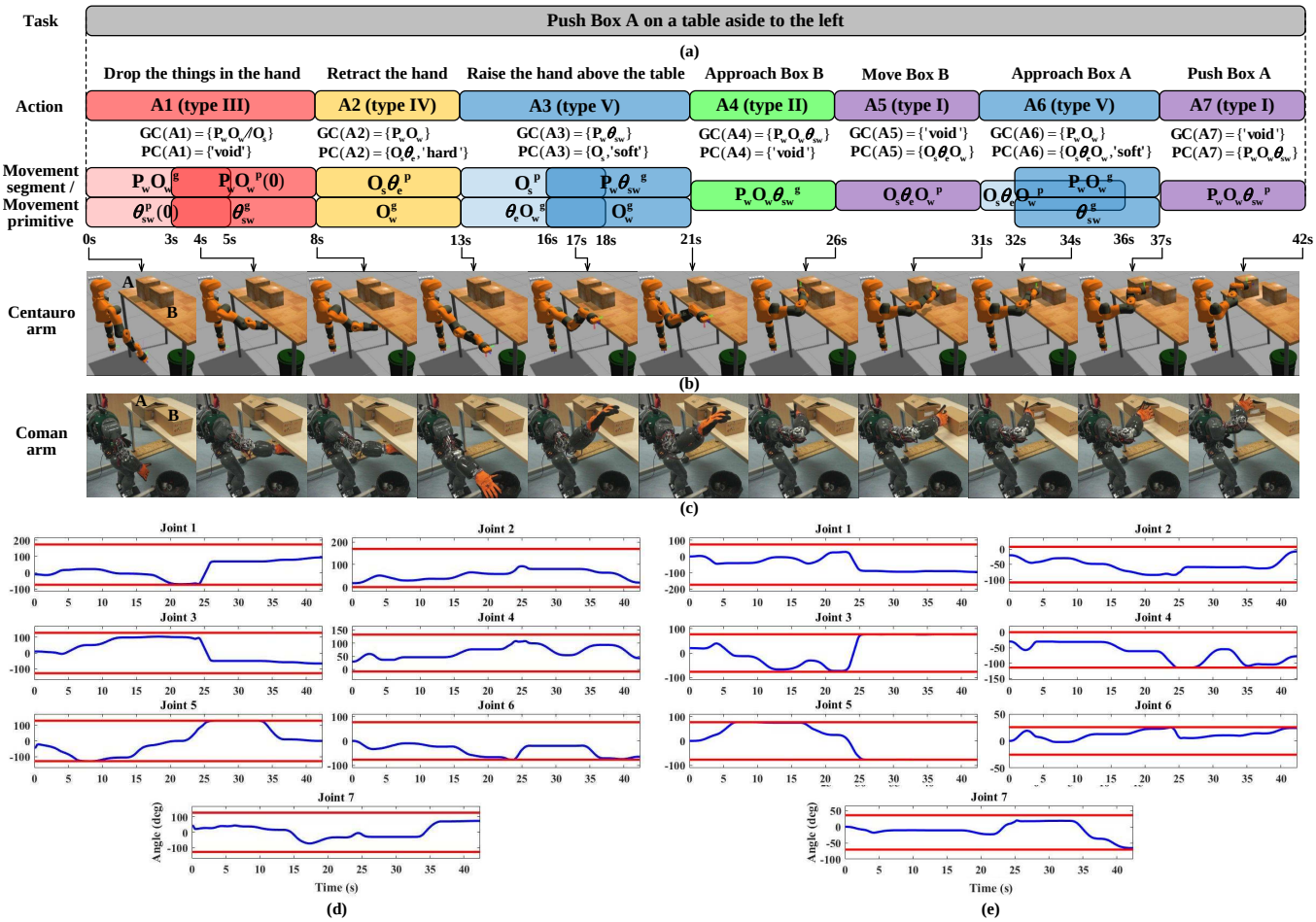

Figure 13: Implementation of a specific task of pushing a box aside on the two different arm platforms by using the proposed A^{2}ML method. (a) The motion structure of the task described in A^{2}ML. Snapshots of the motions of the virtual Centauro right arm (b) and the real Coman right arm (c). The joint trajectories of the Centauro arm (d) and the Coman arm (e). Red lines stand for the joint limits.

to the same final configuration can be produced conveniently, which highly enriches the action design.

3. *Shift of the movement focus of interest*: Each component movement segment of an action possesses a featured goal or path constraint. This featured constraint can be regarded as a MFoI during the period of the movement segment. Since the constraint is usually described by part of the local state variables in the two representation spaces, the MFoI shift during an action or a whole task actually realizes a perfect balance between the motion process control and the adaptation to some hard constraints by using the spare DoFs of the arm.

4. *Powerful and flexible capability of solving for joint trajectories*: Equipped with two distinct methods, i.e., QP-based method and sampling-based method, our framework is able to solve for the joint trajectories of the parameterized movement segment with the satisfaction of different needs in various situations. Since the proposed framework is open, even more methods can be integrated easily.

5. *Generality of the framework*: Benefit from the generality of the two representation spaces of movement primitive and the two methods of solving for the joint trajectories, the proposed A^{2}ML framework can be applied to different anthropomorphic arm platforms to perform the same task in a similar motion "shape" while respecting their mechanical joint angle/velocity limits.

7. Discussion

In Section 6.4, motion and skill transfer was demonstrated between two different anthropomorphic arms. Since the anthropomorphic arms discussed in this paper refer to the robot arms, which resemble the human arm in terms of the shoulder-elbow-wrist configuration and the total degrees of freedom, it is also appealing to explore the possibility of the skill transfer between human arm and anthropomorphic arm based on these physical similarities. That is, using A^{2}ML to describe, analyze and understand human arm motion first and then reconstructing the same motion on an anthropomorphic arm to realize skill transfer. Furthermore, it is worth discussing the reusability of the designed movement primitives and the whole A^{2}ML for more general robot arms, particularly those with less than seven degrees of freedom, for instance, 6-DoF UNIVERSAL UR5 robot arm. Because the movement primitives were designed based on the human arm kinematic structure and human arm motion rules, some of movement primitives are therefore not available

to UR5. For instance, $\boldsymbol{O}_s^p$ can be used to describe the primitive of the upper arm self-rotation, which can not be implemented on UR5 due to the lack of the corresponding degree of freedom. In addition, the reduced number of degrees of freedom make UR5 degenerate into a non-redundant arm, which implies that its configuration will be uniquely determined once the position and orientation of the end-effector are specified. Owing to this change, the complete motion transfer using A^{2}ML from a 7-DoF anthropomorphic arm to a 6-DoF general robot arm is difficult to realize in general.

8. Conclusion

For anthropomorphic arms, a special group of redundant manipulators, a novel and unified task-motion planning framework, A^{2}ML, was proposed and designed according to six important human arm motion hypotheses/rules extracted from the literature in neurophysiology. The A^{2}ML is constructed with four levels and contains a general library of movement primitives and the corresponding motion grammar for organizing them. The effectiveness and performance of this framework were verified by extensive simulations and experiments.

On one hand, thanks to the generality of the designed movement primitives and motion grammar, the A^{2}ML can be applied to different anthropomorphic arms. On the other hand, by means of the variety of movement primitives and the modular feature of the framework, the proposed method is able to effectively solve various task-motion planning problems by organizing elementary movement primitives appropriately. That is, the two diversity issues of the platforms and the complicated tasks can be both resolved in one A^{2}ML framework, which is therefore endowed with the desirable advantage in generality.

Appendix A. Transitional connection between neighboring movment segments within an action

Assume movement segments 1 and 2 are connected in a transitional manner. t_e^1 denotes the ending time of segment 1 and t_s^2 indicates the starting time of segment 2. $q_i^1(t)$ and $q_i^2(t)$ means the joint trajectories of joint i in segments 1 and 2 before connection respectively, and q_i^{12} refers to the joint trajectory of joint i after connection ($i = (1,2,...,7)$). We have:

$$q_i^{12} = \begin{cases} q_i^1(t) & \text{if } t < t_s^2 \\ \beta q_i^1(t) + (1-\beta) q_i^2(t) & \text{if } t_s^2 \le t \le t_e^1 \\ q_i^2(t) & \text{if } t > t_e^1 \end{cases} \tag{A.1}$$

$$\beta = 0.5cos(\frac{t-t_s^2}{t_e^1 - t_s^2}\pi) + 0.5.$$

Appendix B. Detailed derivation of $\boldsymbol{J}_{sw}(\boldsymbol{Q}^{'})$

In this appendix, the calculation of the change rate of the swivel angle, $\theta_{sw}^{'}$, and the corresponding derivation of $\boldsymbol{J}_{sw}(\boldsymbol{Q}^{'})$ in (8) are going to be elaborated. Assume the unit normal vector

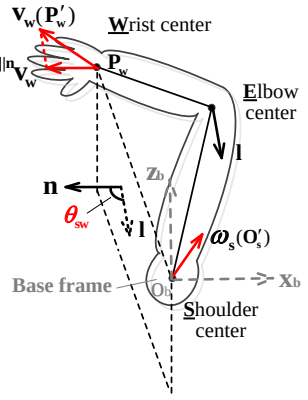

Figure B.14: Schematic diagram of the derivation of the swivel angle derivative.

of the vertical plane, which the center of the wrist joint is in, is $\boldsymbol{n}$ as shown in Fig. B.14. It can be calculated as:

$$\boldsymbol{n} = (\boldsymbol{P}_w \times (-\boldsymbol{z}_b))/\|\boldsymbol{P}_w \times (-\boldsymbol{z}_b)\|. \tag{B.1}$$

According to the defintion, the angle between $\boldsymbol{n}$ and $\boldsymbol{l}$ is the swivel angle, θ_{sw}, and its time derivative can be calculated:

$$\theta_{sw}^{'} = (\arccos(\boldsymbol{n}\cdot\boldsymbol{l}))^{'} = -\frac{(\boldsymbol{n}'\cdot\boldsymbol{l}+\boldsymbol{n}\cdot\boldsymbol{l}')}{\sqrt{1-(\boldsymbol{n}\cdot\boldsymbol{l})^2}}. \tag{B.2}$$

The time derivative of $\boldsymbol{n}$ is caused only by the rotation motion of the vertical plane (represented by $\boldsymbol{n}$) about $\boldsymbol{z}_b$, which is induced by the motion of the wrist center, more specifically, the component of the linear velocity of the wrist along the direction of $\boldsymbol{n}$, $^{\boldsymbol{n}}\boldsymbol{v}_w$. On the other side, the time derivative of $\boldsymbol{l}$ is yielded only by the rotation of shoulder joint, $\boldsymbol{\omega}_s$. Given that the derivative of a vector, caused by a rotation, can be calculated by the cross product of the angular velocity vector of the rotation and the vector, we can derive further:

$$\begin{aligned} -\frac{(\boldsymbol{n}'\cdot\boldsymbol{l}+\boldsymbol{n}\cdot\boldsymbol{l}')}{\sqrt{1-(\boldsymbol{n}\cdot\boldsymbol{l})^2}} &= -\frac{1}{\sqrt{1-(\boldsymbol{n}\cdot\boldsymbol{l})^2}}((\frac{\boldsymbol{v}_w\cdot\boldsymbol{n}}{\sqrt{{}^x\boldsymbol{P}_w^2+{}^y\boldsymbol{P}_w^2}}\boldsymbol{z}_b\times\boldsymbol{n})\cdot\boldsymbol{l} + \\ &\quad \boldsymbol{n}\cdot(\boldsymbol{\omega}_s\times\boldsymbol{l})) \\ &= -\frac{1}{\sqrt{1-(\boldsymbol{n}\cdot\boldsymbol{l})^2}}((\frac{(\boldsymbol{J}_w\boldsymbol{Q}^{'})\cdot\boldsymbol{n}}{\sqrt{{}^x\boldsymbol{P}_w^2+{}^y\boldsymbol{P}_w^2}}\boldsymbol{z}_b\times\boldsymbol{n})\cdot\boldsymbol{l} + \\ &\quad \boldsymbol{n}\cdot((\boldsymbol{J}_s\boldsymbol{Q}^{'})\times\boldsymbol{l})), \end{aligned} \tag{B.3}$$

where ${}^x\boldsymbol{P}_w$ and ${}^y\boldsymbol{P}_w$ stand for the x- and y- coordinates of the center of the wrist respectively. $\boldsymbol{J}_w$ and $\boldsymbol{J}_s$ are the Jabocians which map the joint velocity vector $\boldsymbol{Q}^{'}$ to $\boldsymbol{v}_w$ and $\boldsymbol{\omega}_s$ respectively. They are defined as $\boldsymbol{J}_w = \boldsymbol{J}_{ge}(1:3,1:7)$ and $\boldsymbol{J}_s = [\boldsymbol{J}_{ge}(4:6,1:3)\ \boldsymbol{0}_{3\times4}]$. Utilizing the distributivity of the dot product and the cross product, we can get:

$$\begin{aligned} \theta_{sw}^{'} &= \boldsymbol{J}_{sw}\boldsymbol{Q}^{'}, \\ \boldsymbol{J}_{sw} &= [J_{sw_1}, J_{sw_2}, ..., J_{sw_7}], \\ J_{sw_i} &= -\frac{1}{\sqrt{1-(\boldsymbol{n}\cdot\boldsymbol{l})^2}}((\frac{\boldsymbol{J}_w(:,i)\cdot\boldsymbol{n}}{\sqrt{{}^x\boldsymbol{P}_w^2+{}^y\boldsymbol{P}_w^2}}\boldsymbol{z}_b\times\boldsymbol{n})\cdot\boldsymbol{l} + \\ &\quad \boldsymbol{n}\cdot(\boldsymbol{J}_s(:,i)\times\boldsymbol{l}))\ (i=1,2,...7), \end{aligned} \tag{B.4}$$

where $\boldsymbol{J}_w(:,i)$ and $\boldsymbol{J}_s(:,i)$ denote the ith columns of $\boldsymbol{J}_w$ and $\boldsymbol{J}_s$, respectively.

References

[1] B. R. Duffy, Anthropomorphism and the social robot, Robotics and autonomous systems 42 (3-4) (2003) 177–190.

[2] A. Singh, Robotics as a future and emerging technology, Int. J. Eng. Res. Appl 2 (2012) 156–188.

[3] A. M. Zanchettin, L. Bascetta, P. Rocco, Achieving humanlike motion: Resolving redundancy for anthropomorphic industrial manipulators, IEEE Robotics & Automation Magazine 20 (4) (2013) 131–138.

[4] C. Breazeal, Emotion and sociable humanoid robots, International Journal of Human-Computer Studies 59 (1-2) (2003) 119–155.

[5] E. Garcia, M. A. Jimenez, P. G. De Santos, M. Armada, The evolution of robotics research, IEEE Robotics & Automation Magazine 14 (1) (2007) 90–103.

[6] C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, B. Blumberg, Learning from and about others: Towards using imitation to bootstrap the social understanding of others by robots, Artificial life 11 (1-2) (2005) 31–62.

[7] C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, D. Chilongo, Tutelage and collaboration for humanoid robots, International Journal of Humanoid Robotics 1 (02) (2004) 315–348.

[8] K. Macherey, A. M. Dai, D. Talbot, A. C. Popat, F. Och, Language-independent compound splitting with morphological operations, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, Association for Computational Linguistics, 2011, pp. 1395–1404.

[9] A. M. Dai, Q. V. Le, Semi-supervised sequence learning, in: Advances in neural information processing systems, 2015, pp. 3079–3087.

[10] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, S. Bengio, Generating sentences from a continuous space, arXiv preprint arXiv:1511.06349.

[11] A. Liegeois, Automatic supervisory control of the configuration and behavior of multibody mechanisms, IEEE transactions on systems, man, and cybernetics 7 (12) (1977) 868–871.

[12] N. Mansard, O. Stasse, P. Evrard, A. Kheddar, A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks, in: Advanced Robotics, 2009. ICAR 2009. International Conference on, IEEE, 2009, pp. 1–6.

[13] A. Escande, N. Mansard, P.-B. Wieber, Hierarchical quadratic programming: Fast online humanoid-robot motion generation, The International Journal of Robotics Research 33 (7) (2014) 1006–1028.

[14] E. Spyrakos-Papastavridis, J. S. Dai, P. R. Childs, N. G. Tsagarakis, Selective-compliance-based lagrange model and multilevel noncollocated feedback control of a humanoid robot, Journal of Mechanisms and Robotics 10 (3) (2018) 031009.

[15] M. Li, R. Kang, D. T. Branson, J. S. Dai, Model-free control for continuum robots based on an adaptive kalman filter, IEEE/ASME Trans. Mechatron 23 (1) (2018) 286–297.

[16] J. J. Kuffner, S. M. LaValle, Rrt-connect: An efficient approach to single-query path planning, in: Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on, Vol. 2, IEEE, 2000, pp. 995–1001.

[17] N. Ratliff, M. Zucker, J. A. Bagnell, S. Srinivasa, Chomp: Gradient optimization techniques for efficient motion planning, in: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, IEEE, 2009, pp. 489–494.

[18] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal, Stomp: Stochastic trajectory optimization for motion planning, in: Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 4569–4574.

[19] H. Liu, J. Dai, An approach to carton-folding trajectory planning using dual robotic fingers, Robotics and Autonomous Systems 42 (1) (2003) 47–63.

[20] W. Yao, J. S. Dai, Dexterous manipulation of origami cartons with robotic fingers based on the interactive configuration space, Journal of Mechanical Design 130 (2) (2008) 022303.

[21] W. Yao, F. Cannella, J. S. Dai, Automatic folding of cartons using a reconfigurable robotic system, Robotics and Computer-Integrated Manufacturing 27 (3) (2011) 604–613.

[22] R. E. Fikes, N. J. Nilsson, Strips: A new approach to the application of theorem proving to problem solving, Artificial intelligence 2 (3-4) (1971) 189–208.

[23] F. Gravot, S. Cambon, R. Alami, asymov: a planner that deals with intricate symbolic and geometric problems, Robotics Research, The Eleventh International Symposium (2005) 100–110.

[24] S. Cambon, R. Alami, F. Gravot, A hybrid approach to intricate motion, manipulation and task planning, The International Journal of Robotics Research 28 (1) (2009) 104–126.

[25] L. P. Kaelbling, T. Lozano-Pérez, Integrated task and motion planning in belief space, The International Journal of Robotics Research 32 (9-10) (2013) 1194–1227.

[26] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, P. Abbeel, Combined task and motion planning through an extensible planner-independent interface layer, in: Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, 2014, pp. 639–646.

[27] P. Viviani, Do units of motor action really exist, Experimental Brain Research 15 (1986) 201–215.

[28] F. A. Mussa-Ivaldi, E. Bizzi, Motor learning through the combination of primitives, Philosophical Transactions of the Royal Society of London B: Biological Sciences 355 (1404) (2000) 1755–1769.

[29] A. d'Avella, P. Saltiel, E. Bizzi, Combinations of muscle synergies in the construction of a natural motor behavior, Nature neuroscience 6 (3) (2003) 300–308.

[30] A. J. Ijspeert, J. Nakanishi, S. Schaal, Movement imitation with nonlinear dynamical systems in humanoid robots, in: Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on, Vol. 2, IEEE, 2002, pp. 1398–1403.

[31] P. Pastor, M. Kalakrishnan, F. Meier, F. Stulp, J. Buchli, E. Theodorou, S. Schaal, From dynamic movement primitives to associative skill memories, Robotics and Autonomous Systems 61 (4) (2013) 351–361.

[32] H. B. Amor, G. Neumann, S. Kamthe, O. Kroemer, J. Peters, Interaction primitives for human-robot cooperation tasks, in: Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, 2014, pp. 2831–2837.

[33] M. Ewerton, G. Neumann, R. Lioutikov, H. B. Amor, J. Peters, G. Maeda, Learning multiple collaborative tasks with a mixture of interaction primitives, in: Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, 2015, pp. 1535–1542.

[34] I. V. Grinyagin, E. V. Biryukova, M. A. Maier, Kinematic and dynamic synergies of human precision-grip movements, Journal of neurophysiology 94 (4) (2005) 2284–2294.

[35] F. Polyakov, E. Stark, R. Drori, M. Abeles, T. Flash, Parabolic movement primitives and cortical states: merging optimality with geometric invariance, Biological cybernetics 100 (2) (2009) 159.

[36] Y. Li, T. Wang, H.-Y. Shum, Motion texture: a two-level statistical model for character motion synthesis, in: ACM Transactions on Graphics (ToG), Vol. 21, ACM, 2002, pp. 465–472.

[37] C. Rose, M. F. Cohen, B. Bodenheimer, Verbs and adverbs: Multidimensional motion interpolation, IEEE Computer Graphics and Applications 18 (5) (1998) 32–40.

[38] G. Guerra-Filho, Y. Aloimonos, A language for human action, Computer 40 (5).

[39] C. Fang, X. Ding, A unified language for anthropomorphic arm motion, in: Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on, IEEE, 2013, pp. 522–529.

[40] N. Wirth, What can we do about the unnecessary diversity of notation for syntactic definitions?, Communications of the ACM 20 (11) (1977) 822–823.

[41] T. Flash, B. Hochner, Motor primitives in vertebrates and invertebrates, Current opinion in neurobiology 15 (6) (2005) 660–666.

[42] M. Desmurget, S. Grafton, Forward modeling allows feedback control for fast reaching movements, Trends in cognitive sciences 4 (11) (2000) 423–431.

[43] C. A. Buneo, M. R. Jarvis, A. P. Batista, R. A. Andersen, Direct visuomotor transformations for reaching, Nature 416 (6881) (2002) 632–636.

[44] R. A. Scheidt, J. B. Dingwell, F. A. Mussa-Ivaldi, Learning to move amid uncertainty, Journal of neurophysiology 86 (2) (2001) 971–985.

[45] S. Berman, D. G. Liebermann, J. McIntyre, Constrained motion control on a hemispherical surface: path planning, Journal of neurophysiology 111 (5) (2014) 954–968.

[46] C. Zhou, C. Fang, X. Wang, Z. Li, N. Tsagarakis, A generic optimization-based framework for reactive collision avoidance in bipedal locomotion, in: Automation Science and Engineering (CASE), 2016 IEEE Interna-

tional Conference on, IEEE, 2016, pp. 1026–1033.
[47] G. Oriolo, C. Mongillo, Motion planning for mobile manipulators along given end-effector paths, in: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, IEEE, 2005, pp. 2154–2160.
[48] M. Rushworth, H. Johansen-Berg, S. M. Göbel, J. Devlin, The left parietal and premotor cortices: motor attention and selection, Neuroimage 20 (2003) S89–S100.
[49] M. F. Rushworth, A. Ellison, V. Walsh, Complementary localization and lateralization of orienting and motor attention., Nature neuroscience 4 (6).
[50] T. Flash, N. Hogan, The coordination of arm movements: an experimentally confirmed mathematical model, Journal of neuroscience 5 (7) (1985) 1688–1703.
[51] N. Hogan, An organizing principle for a class of voluntary movements, Journal of Neuroscience 4 (11) (1984) 2745–2754.
[52] D. Tolani, A. Goswami, N. I. Badler, Real-time inverse kinematics techniques for anthropomorphic limbs, Graphical models 62 (5) (2000) 353–388.
[53] C. Fang, A. Rocchi, E. M. Hoffman, N. G. Tsagarakis, D. G. Caldwell, Efficient self-collision avoidance based on focus of interest for humanoid robots, in: Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on, IEEE, 2015, pp. 1060–1066.
[54] C. Fang, X. Ding, A global obstacle-avoidance map for anthropomorphic arms, International Journal of Advanced Robotic Systems 11 (7) (2014) 117.
[55] J. Nakanishi, R. Cory, M. Mistry, J. Peters, S. Schaal, Operational space control: A theoretical and empirical comparison, The International Journal of Robotics Research 27 (6) (2008) 737–757.
[56] C. Fang, J. Lee, A. Ajoudani, C. Zhou, N. G. Tsagarakis, D. G. Caldwell, RRT-based motion planning with sampling in redundancy space for robots with anthropomorphic arms, in: Advanced Intelligent Mechatronics (AIM), 2017 IEEE International Conference on, IEEE, 2017, pp. 1612–1618.
[57] X. Ding, C. Fang, A motion planning method for an anthropomorphic arm based on movement primitives of human arm triangle, in: Mechatronics and Automation (ICMA), 2012 International Conference on, IEEE, 2012, pp. 303–310.
[58] X. Ding, C. Fang, A novel method of motion planning for an anthropomorphic arm based on movement primitives, IEEE/ASME Transactions on Mechatronics 18 (2) (2013) 624–636.
[59] L. Baccelliere, N. Kashiri, L. Muratore, A. Laurenzi, M. Kamedula, A. Margan, S. Cordasco, J. Malzahn, N. G. Tsagarakis, Development of a human size and strength compliant bi-manual platform for realistic heavy manipulation tasks, in: Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on, IEEE, 2017, pp. 5594–5601.
[60] N. G. Tsagarakis, S. Morfey, G. M. Cerda, L. Zhibin, D. G. Caldwell, Compliant humanoid coman: Optimal joint stiffness tuning for modal frequency control, in: Robotics and Automation (ICRA), 2013 IEEE International Conference on, IEEE, 2013, pp. 673–678.