

IMECE2014-37700

## A GENERAL METHOD FOR KINEMATIC RETARGETING: ADAPTING POSES BETWEEN HUMANS AND ROBOTS

**Tarik Tosun\***

GRASP Laboratory  
Department of Mechanical Engineering  
and Applied Mechanics  
University of Pennsylvania  
Philadelphia, Pennsylvania 19104  
Email: tarikt@seas.upenn.edu

**Ross Mead**

Interaction Lab  
Department of Computer Science  
University of Southern California  
Los Angeles, California, 90089  
rossmead@usc.edu

**Robert Stengel**

Department of Mechanical  
and Aerospace Engineering  
Princeton University  
Princeton, New Jersey, 08544  
stengel@princeton.edu

### ABSTRACT

This paper presents a method for kinematic retargeting that is general to a broad class of kinematic chains. *Kinematic retargeting* is the adaptation of a pose or motion from one kinematic embodiment to another. Our method distinguishes itself in its ability to adapt poses to new robots with very little configuration by the user. We accomplish this by defining two general metrics for retargeting and minimizing a cost function which is the weighted sum of these two metrics. This allows the method to automatically adapt poses between source and target chains that have different link lengths and degrees of freedom.

These capabilities address a specific problem in Human-Robot Interaction (HRI), where behaviors are often defined in a robot-specific manner. The ability to automatically adapt behaviors from humans to new robots, and from one robot to another, will facilitate experimental repeatability. Through simulation and experiments, we demonstrate that our method is effective in adapting poses across chains with different numbers of joints, and in adapting socially expressive gestures from a human to two very different robots.

### 1 Introduction

In this paper, we present a method for *kinematic retargeting*, the adaptation of a pose or motion from one kinematic embodiment (the *source*) to another (the *target*). Our method was de-

veloped for use in situations where either the target or both the source and target are robots. Poses may be adapted between any two serial chains with revolute joints, even if they have different link lengths, numbers of joints, or degrees of freedom. Our goal is to enable the transfer of gesture behaviors from humans to robots, and the transfer of gesture libraries across different robots, with a focus on applications in Human-Robot Interaction (HRI).

#### 1.1 Motivation: Human-Robot Interaction

The essential question behind kinematic retargeting is: what constitutes a good adaptation of a pose or motion? The answer to this question depends on the context. If we want to retarget the motions of a ballet dancer, for example, the lines and angles formed by her arms, legs, and torso might be most important, whereas for a tap dancer we might only be concerned with the position of her feet and the rhythm with which they strike the floor. Because of the enormous variety of [potentially very complex] factors that might influence the quality of behavior adaptation in a particular context, developing a retargeting method that is general with respect to all contexts would be extremely difficult. A more tractable problem is to develop a retargeter for a specific context (for specific kinds of behaviors) that is general to a broad class of kinematic chains, allowing it to be used with a wide variety of robots.

The context we focus on in this paper is the adaptation of socially expressive gestures from human sources to robot targets,

---

\*Address all correspondence to this author.

and from robot sources to robot targets. These capabilities address a specific problem in Human-Robot Interaction (HRI), in which robots use socially expressive gestures (often coordinated with speech) to communicate with people [1], [2]. For each HRI scenario, a set of embodied robot behaviors must be defined (or learned). The behaviors are often defined in a platform-specific manner, as in any field of robotics or engineering more generally. When a library of gestures is developed for one robot, differences in size, actuation, and proportion with other robot platforms will make it difficult to directly apply this library to other platforms. Similarly, if motion capture is used to generate a new gesture from human demonstration, it is challenging to map the behavior directly onto a robot. This inhibits experimental repeatability, especially in a relatively young field where social robot platforms are not standardized.

In this paper, we present a method that addresses these issues. The ability to retarget gestures from a human motion capture system to an arbitrary robot can be used to define new gestural behaviors by example. Perhaps more importantly, the ability to retarget from one robot to another enables the transference of behavior libraries between different robot platforms.

The larger aim of this project is toward a general framework for the transference of social behaviors between very different robot platforms. Such a framework would be of great utility in the HRI community, as it would facilitate a universal library of experimentally validated, platform-independent robot behaviors. A unified repository of experimentally validated behaviors, coupled with a method for automatically adapting those behaviors to specific robot platforms, would allow greater standardization and repeatability in the field.

## 1.2 Introduction to the Method

This paper presents a method to retarget poses and motion between any two serial kinematic chains with revolute joints, even if they vary in link lengths or degrees of freedom. We accomplish this by treating the pose of a kinematic chain in a geometric sense, defining it as a piecewise linear function that traces the links of a scaled form of the chain. This allows us to define a metric of pose similarity that follows naturally from the typical notion of fitting error between parametric curves, and to minimize the error in a least-squares sense. We define the normalized distance between the end-effectors of two chains as a second metric. This decision was based on evidence that end-effector position is of special importance in socially expressive gesturing [3]. At the core of our retargeting method is a numerical solver that minimizes a weighted sum of these two metrics. Because the metrics are applicable to all serial chains with revolute joints, the software system we have developed can be easily applied to a wide variety of robots.

Our approach makes sacrifices to achieve generality. We tell our solver very little about the desirable properties of the source

motion, and make no attempt to model its semantic intent (ie: pointing to an external referent). The advantage of our method is that it can adapt poses to a new robot with almost no configuration. The user only needs to specify the kinematic structure of the new robot and a single parameter (the relative weighting of the two metrics). Our premise is that by generating pose adaptations that are similar to the source motion in a geometric sense at a sufficiently high frame rate, we will preserve the gross characteristics of the source behaviors without explicitly modeling them. For applications in HRI, this will serve to appropriately adapt socially expressive gestures between humans and robots.

## 2 Related Work

Existing methods for retargeting can be found in the literature of robotics and animation. This section summarizes relevant approaches in three categories: (A) retargeting in animation, (B) configuration-space methods in robotics, and (C) task-space methods in robotics.

### 2.1 Retargeting in Animation

In animation, many retargeting methods focus on adapting motion from one humanoid character to another. Gleicher [4] presents a method for retargeting between humanoids with identical embodiments which preserves certain motion constraints to maintain the realism of the motion. Like our method, Gleicher presents an offline method that optimizes a quadratic objective function, although the nature of our objective functions differ significantly. Gleicher treats the more general problem of retargeting between non-identical morphologies only briefly. He introduces a method that adapts motion from a humanoid source to a non-humanoid target (such as an animated soda can) by imbuing the target with an appropriately sized humanoid kinematic skeleton, and then retargeting to this skeleton. This method works well for animated characters, but is of limited application in robotics because robots have a fixed kinematic structure that cannot be adapted or changed.

Hecker *et al.* [5] present a method for motion retargeting that is designed for use with diverse non-humanoid characters. The method relies on semantic information provided by animators to create generalized animation curves for behaviors, which are then specialized onto a given character using a high-performance IK solver. The method is illustrative rather than example-based, bringing animators into the retargeting loop by asking them to describe important factors *a priori*. While this strategy achieves impressive success in generalizing behaviors between embodiments, it is primarily useful for animators with the specialized knowledge required to create generalizable animation curves. In this paper, we present a method that can automatically port an existing behavior between embodiments, without requiring the user to explicitly model the important semantic factors of the be-

havior.

## 2.2 Configuration-space Retargeting in Robotics

Many retargeting methods in robotics are designed specifically for the teleoperation or instruction of humanoid robots; they are expressed in configuration space (i.e. joint angle space). As humanoid robots typically replicate a subset of human kinematic capabilities, some methods adapt motion by scaling and limiting source motion to capabilities of the target robot [6, 7, 8].

Signal processing techniques have been applied to preserve or highlight desirable features of retargeted motion. Gieliniak [9, 10] defines metrics for “natural” or “human-like” movement and optimizes these metrics to produce high-quality retargeted motions with humanoid robots. These techniques for maximizing human-likeness are applied as a post-processing step after retargeting, and could be used in conjunction with the method we present in this paper.

## 2.3 Task-space Retargeting in Robotics

Dariush *et al.* [11, 12] present a task space method that allows the user to specify a set of Cartesian task descriptors defining correspondences between points on the human body and a humanoid robot. An on-line control framework computes differential kinematics relating task variables and joint variables, and minimizes tracking error. Similarly, Ott *et al.* presents a task space method in which a set of control points on the human body are attached to corresponding points on a humanoid robot via virtual springs which drive a simplified simulation of the robot dynamics [13].

While these system allows some level of generality with respect to tasks, they are not general with respect to the source and target kinematic chains; for kinematically dissimilar sources and targets there may be no obvious choice of correspondences between the two chains. As an example, consider a human source and target arm made of four modular robots (Figure 1). The modular robot arm has four joints evenly spaced along its length, and is kinematically very different from a human arm (kinematic details can be found in Figure 8). Consequently, it is not obvious how to choose points of correspondence between the two arms.

## 3 Kinematic Retargeting

In this section, two metrics for kinematic retargeting are formally defined. Chains consist of a series of linkages with revolute joints, such that each link has a single degree of freedom relative the previous link. In the discussion following, the base link of all chains is assumed to lie at the origin  $([0, 0, 0])$ . Kinematic retargeting is formally posed as follows:

**Given:** Two kinematic chains  $\mathbf{S}$  and  $\mathbf{T}$ , with known kinematics, and a known configuration for  $\mathbf{S}$ .



**FIGURE 1:** Choice of correspondence points between human arm (source) and modular robot arm (target) is not obvious.

**Find:** A configuration for  $\mathbf{T}$  which minimizes the retargeting error  $E$  between  $\mathbf{S}$  and  $\mathbf{T}$ .

Kinematic retargeting is posed as an  $N_T$ -dimensional constrained optimization problem, where  $N_T$  is the number of degrees of freedom of the target chain. Target joint-angle limits define inequality constraints on the domain. Given a known configuration of the source chain  $\mathbf{S}$ , we define the retargeting error  $E$  between  $\mathbf{T}$  and  $\mathbf{S}$  as a function of the configuration  $\underline{\theta}_T$  of  $\mathbf{T}$ . In our framework,  $E$  is the weighted sum of two metrics:  $E_p$ , the *pose error metric* and  $E_e$ , the *end-effector error metric*. Both metrics are defined in terms of the *normalized pose* of the source and target chains, which we describe in Section (A) below. The two metrics are then described in Sections (B) and (C).

### 3.1 Normalized Pose

Consider a kinematic chain  $\mathbf{C}$  (i.e. the source chain,  $\mathbf{S}$ , or the target chain,  $\mathbf{T}$ ) with  $N_C$  joints and links, and whose link lengths are  $\{\ell_1, \ell_2, \dots, \ell_{N_C}\}$ .  $\hat{\mathbf{C}}$  is the *normalized form* of  $\mathbf{C}$ , a chain with joints identical to  $\mathbf{C}$  and but with link lengths  $\{\ell_1/L, \ell_2/L, \dots, \ell_{N_C}/L\}$ , where  $L = \sum_{i=1}^{N_C} \ell_i$ .  $\hat{\mathbf{C}}$  has unit length but is otherwise kinematically identical to  $\mathbf{C}$ .

Given a configuration (set of joint angles)  $\underline{\theta}_C = [\theta_1, \theta_2, \dots, \theta_{N_C}]$  for  $\mathbf{C}$ , the *normalized pose* of  $\mathbf{C}$  is the set of contiguous line segments connecting the joints of  $\hat{\mathbf{C}}$  in configuration  $\underline{\theta}_C$ . It may be expressed as a piecewise linear function

$p(s) : [0, 1] \rightarrow [-1, 1]^3$ , with  $s$  parametrizing length along  $\hat{\mathbf{C}}$  from base to end effector, so that  $p(0)$  returns  $[0, 0, 0]$  (the coordinates of the base joint), and  $p(1)$  returns the cartesian coordinates of the end effector of  $\hat{\mathbf{C}}$ . For a given kinematic chain  $\mathbf{C}$ ,  $P_C(\underline{\theta}_C, s)$  is the normalized pose function, which is  $p(s)$  for  $\mathbf{C}$  in configuration  $\underline{\theta}_C$ . Defining the metrics in terms of normalized pose is advantageous because it makes them dimensionless, allowing them to be applied to chains of different total lengths.

### 3.2 Pose Error Metric

The pose error metric,  $E_p$ , is a measure of fitting error between the geometric shapes defined by the source and target chains. It is defined as the squared distance between differential points on each of their normalized forms, integrated over the [unit] lengths of their normalized forms. This error metric follows naturally from the definition of normalized pose in the sense that the pose of each chain is a parametric curve in  $\mathbb{R}^3$ , and an integrated squared error defined in this way may be interpreted as a residual or fitting error between general parametric curves in  $\mathbb{R}^3$ .

**Definition** The *pose error metric* between two kinematic chains is the squared error between their respective normalized pose functions, integrated lengthwise from base to end-effector.

$$E_p = \int_0^1 \|P_S(\underline{\theta}_S, s) - P_T(\underline{\theta}_T, s)\|^2 ds \quad (1)$$

### 3.3 End-Effector Error Metric

The end-effector error metric,  $E_e$ , represents error in the positions of the end-effectors of the two chains. End-effector position is crucial in socially expressive gesturing: related literature indicates that people attend to the hand or end-effector during gesture interpretation and analysis [3]. This provides a compelling motivation to incorporate an end-effector error metric in the retargeting process. As with  $E_p$ , we define  $E_e$  in terms of the normalized form of the source and target chains.

**Definition** The *end-effector error metric* between two kinematic chains  $\mathbf{S}$  and  $\mathbf{T}$  is the square of the distance between the end-effectors of the normalized forms of the two chains.

$$E_e = \|P_S(\underline{\theta}_S, 1) - P_T(\underline{\theta}_T, 1)\|^2 \quad (2)$$

### 3.4 Overall Error Metric

The retargeting error metric  $E$ , between two chains  $S$  and  $T$  is the weighted sum of  $E_p$  and  $E_e$ :

$$E = E_p + \alpha E_e \quad (3)$$

Here,  $\alpha$  is a weighting coefficient which determines how much priority is placed on each metric. Section 4.2 explores the effect of different values of  $\alpha$ .

## 3.5 Optimization

Our approach poses kinematic retargeting as a constrained optimization problem:

$$\underline{\theta}_T^* = \arg \min_{\underline{\theta}_T} E(\underline{\theta}_T), \text{ subject to } \underline{\theta}_T \in [\underline{\ell}_T, \underline{u}_T] \quad (4)$$

where  $\underline{\ell}_T$  and  $\underline{u}_T$  are the lower and upper angular joint limits that bound the domain of the target configuration  $\underline{\theta}_T$ . This problem formulation is amenable to solution by a variety of numerical optimization methods. An in-depth consideration of available methods is beyond the scope of this paper, but the reader may refer to [14] [15] for more information.

## 4 Validation in Simulation

In this section, we demonstrate the performance of our method in retargeting poses across source and target chains with different degrees of freedom. We then explore the effect of different values of  $\alpha$  on the solutions generated by our method. A MATLAB<sup>®</sup> implementation was used for testing and analysis. Kinematic chains were modeled using Denavit-Hartenberg (DH) parameters, so that the positions of all of a chain's joints could be easily computed given its joint angles. Functions for chain length scaling, numerical integration, and computation of the retargeting metric were also implemented. The pose error metric  $E_p$  (Eq. 1) is approximated in software as a finite sum, where  $\delta = 1/N$ :

$$E_p \approx \sum_{n=1}^N \|P_S(\underline{\theta}_S, n\delta) - P_T(\underline{\theta}_T, n\delta)\|^2 \delta \quad (5)$$

In our implementation,  $N = 100$  was used, but good results can be achieved with meshings as coarse as  $N = 10$  for a 5-link target chain. Optimization is performed using the `fmincon` function, which provides an active-set method for optimization with inequality constraints [16]. A convergence tolerance of 1E-10 was used in our experiments.

### 4.1 Effectiveness across Different DOF

Our method is able to retarget poses between source and target chains with different degrees of freedom (DOF). In the experiments following, source poses are the 2D projection of frames extracted from motion capture of a human arm moving through

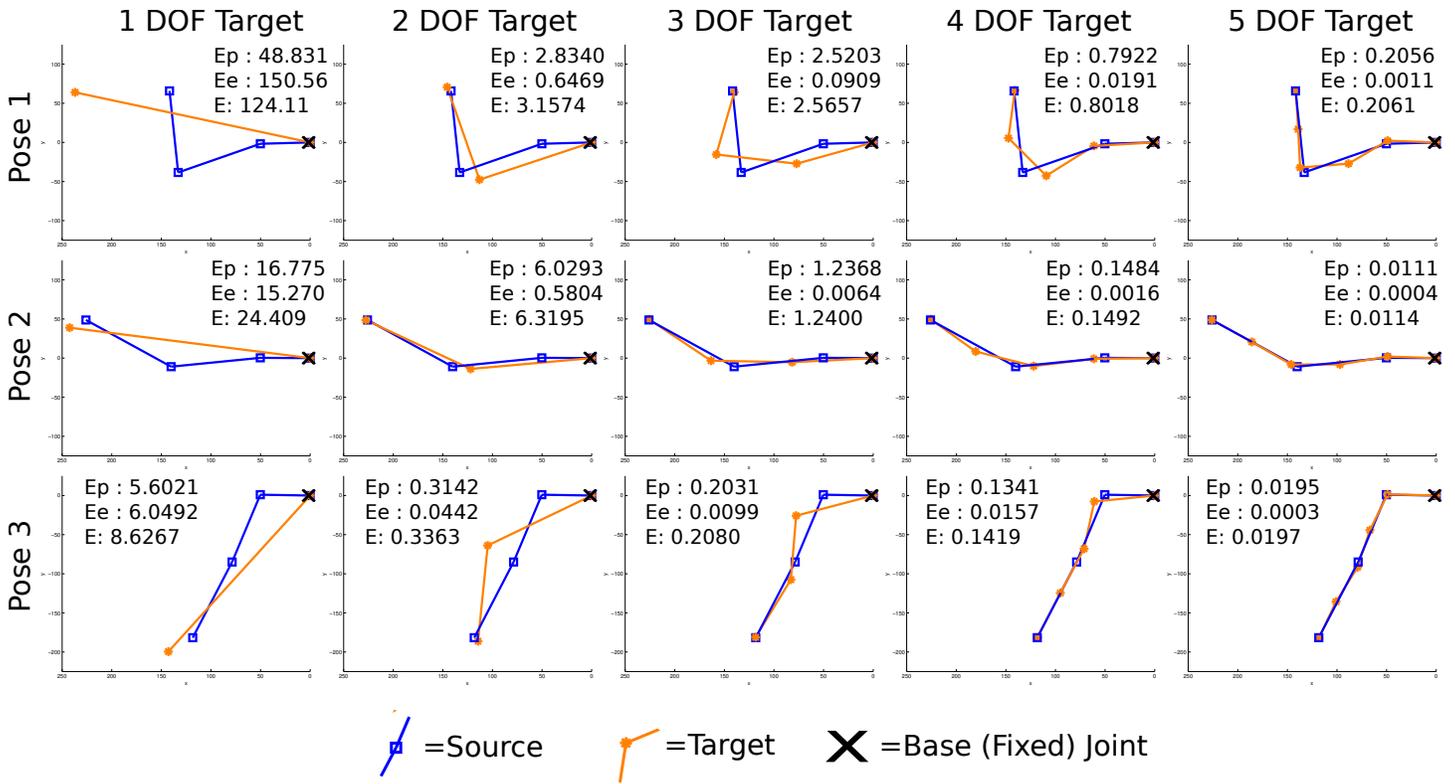


FIGURE 2: 2d Comparison. Metric values shown have been multiplied by  $1E3$  for ease of readability.

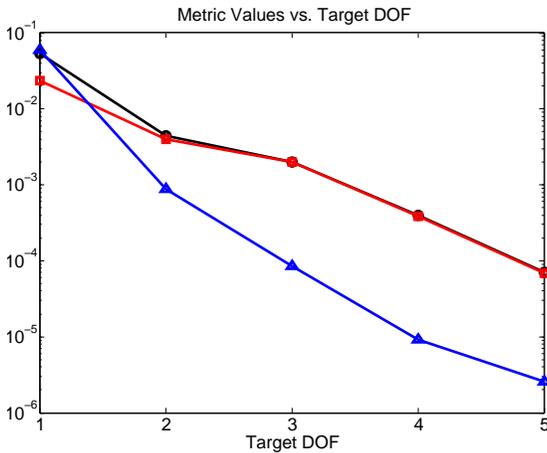


FIGURE 3: Average values of  $E_p$  (red, squares),  $E_e$  (blue, triangles), and  $E$  (black, circles) for retargeted solutions, plotted against target DOF.

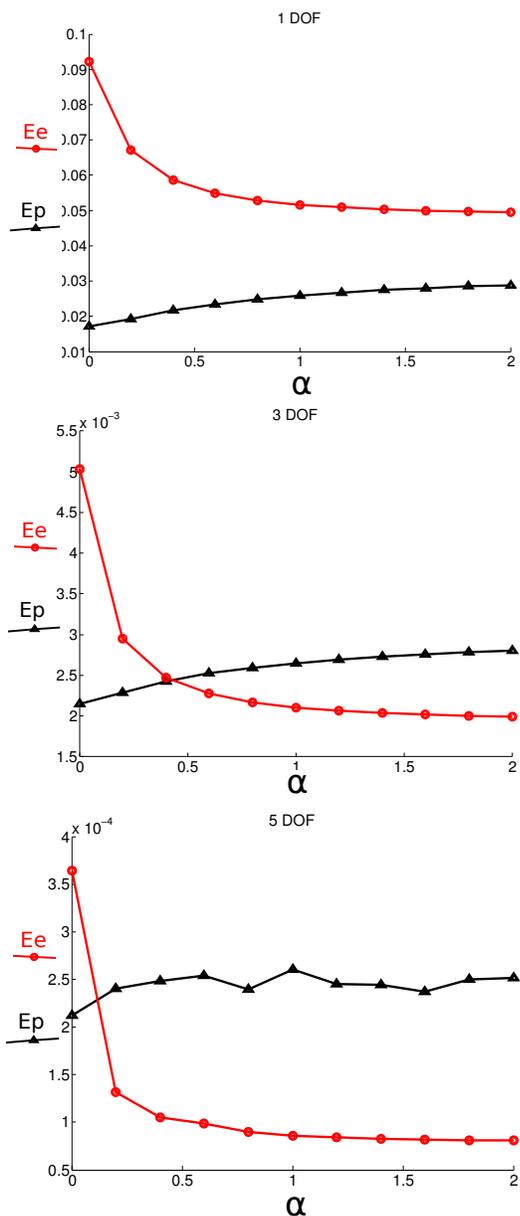
its entire range of motion in the coronal plane (parallel to the chest and dividing the body in half front to back). More detail on our software system and motion capture techniques can be found in Section 5. It is important to note that in our method a kine-

matic model of the source is not required: if the positions of the source joints are known (e.g. from motion capture), the source normalized pose may be obtained directly by scaling the set of line segments defined by the source joints to unit total length.

Poses from the 3-link source chain were retargeted to target chains with one through five equally spaced joints. The target chains are of the same total length as the source, so that poses can be easily compared. In these tests,  $\alpha = 0.5$  was used. Figure 2 provides a visualization of retargeting results for three frames of the coronal plane data set. The source chain (blue, with squares representing joints) and target chain (orange, with asterisks representing joints) are overlaid so that the quality of the solutions can be visually inspected. Figure 3 plots average values of  $E_p$ ,  $E_e$ , and  $E$  for retargeting solutions from twenty-five frames of the coronal plane data set to the 1-5 DOF targets.

In Figure 3, average values of  $E_p$ ,  $E_e$ , and  $E$  all monotonically decrease as the number of joints of the target chain increases. This is intuitive: we should expect that a chain with fewer degrees of freedom would not be able to approximate the pose of one with more degrees of freedom perfectly. The three and four joint targets have as many or more degrees of freedom than the source, but are still unable to exactly replicate the source pose because they have different segment lengths than the source. As a result, the additional flexibility afforded the four joint tar-

get allow it to assume poses closer to the source than the three link target. In Figure 2, we see that the 5-DOF target is proportioned so that two of its joints are almost perfectly aligned with the joints of the source. Consequently, retargeted solutions for the 5-DOF target lie nearly tangent to the source, and  $E$  values are very low (below  $1E - 4$  on average).



**FIGURE 4:** Plots of  $E_p$  (black, triangles) and  $E_e$  (red, circles) versus  $\alpha$  for 1, 3, and 5 DOF targets retargeted from 3 DOF source. We see diminishing returns for  $\alpha > 0.5$ .

## 4.2 Effect of $\alpha$

The weighting coefficient  $\alpha$  controls the relative importance of the two metrics in retargeting solutions. Figure 4 shows plots of  $E_p$  and  $E_e$  versus  $\alpha$  for retargeted solutions from the 3-DOF source to the 1-, 3-, and 5-DOF targets. For all three target chains, we see a sharp initial decrease in  $E_e$  as  $\alpha$  is increased, but diminishing returns after we increase  $\alpha$  beyond a certain point. Beyond the point of diminishing returns, there is a trade-off between  $E_p$  and  $E_e$ , and the solution can be tuned for better pose fit or end effector matching by varying  $\alpha$ . Beyond the point of diminishing returns, there is no trade-off: increasing  $\alpha$  is detrimental to  $E_p$  and has little effect on  $E_e$ .

The value of  $\alpha$  beyond which we begin to see diminishing returns depends on the nature of the source and target chains. When the target chain has degrees of freedom that allow it to replicate or subsume the capabilities of the source, target configurations which closely match both the pose and end-effector position may be found for most source poses. Consequently, the region of trade-off between  $E_e$  and  $E_p$  will be small, and diminishing returns will begin at a low value of  $\alpha$ . The 5-DOF target, for example, shows diminishing returns for  $\alpha$  values greater than 0.4. When the target chain cannot subsume the capabilities of the source chain, it will be unable to match both the pose and the end effector perfectly, and there will be a larger region of trade-off between  $E_e$  and  $E_p$ . This can be seen in the 1-DOF target, which does not show diminishing returns until an  $\alpha$  value of 1.

When the target chain is able to subsume the capabilities of the source, low values of  $\alpha$  are preferable. However, because  $E_p$  and  $E_e$  tend to be very low in these cases, increasing  $\alpha$  beyond the point of diminishing returns has little qualitative effect, and in general solutions are relatively insensitive to  $\alpha$ . When the target chain cannot subsume the capabilities of the source, the region of trade-off extends over a larger range of  $\alpha$  values, and  $\alpha$  can be adjusted to tune solutions for better pose fit or end-effector matching. In practice, we have found that  $\alpha = 0.5$  is a good starting point for most problems.

## 5 Validation with Physical Robots

The MATLAB<sup>®</sup> implementation of our method described in Sec. 4 was used to retarget human motion capture data to two robots. Markerless motion capture was performed using the Microsoft Kinect and the open-source OpenNI and PrimeSense NITE libraries for Ubuntu Linux [17] [18]. To access OpenNI and NITE capabilities from within MATLAB<sup>®</sup>, the Kinect\_Matlab package was used [19]. Robot kinematic models, shown in Figure 8, were built in MATLAB so that forward kinematics could be computed during optimization.

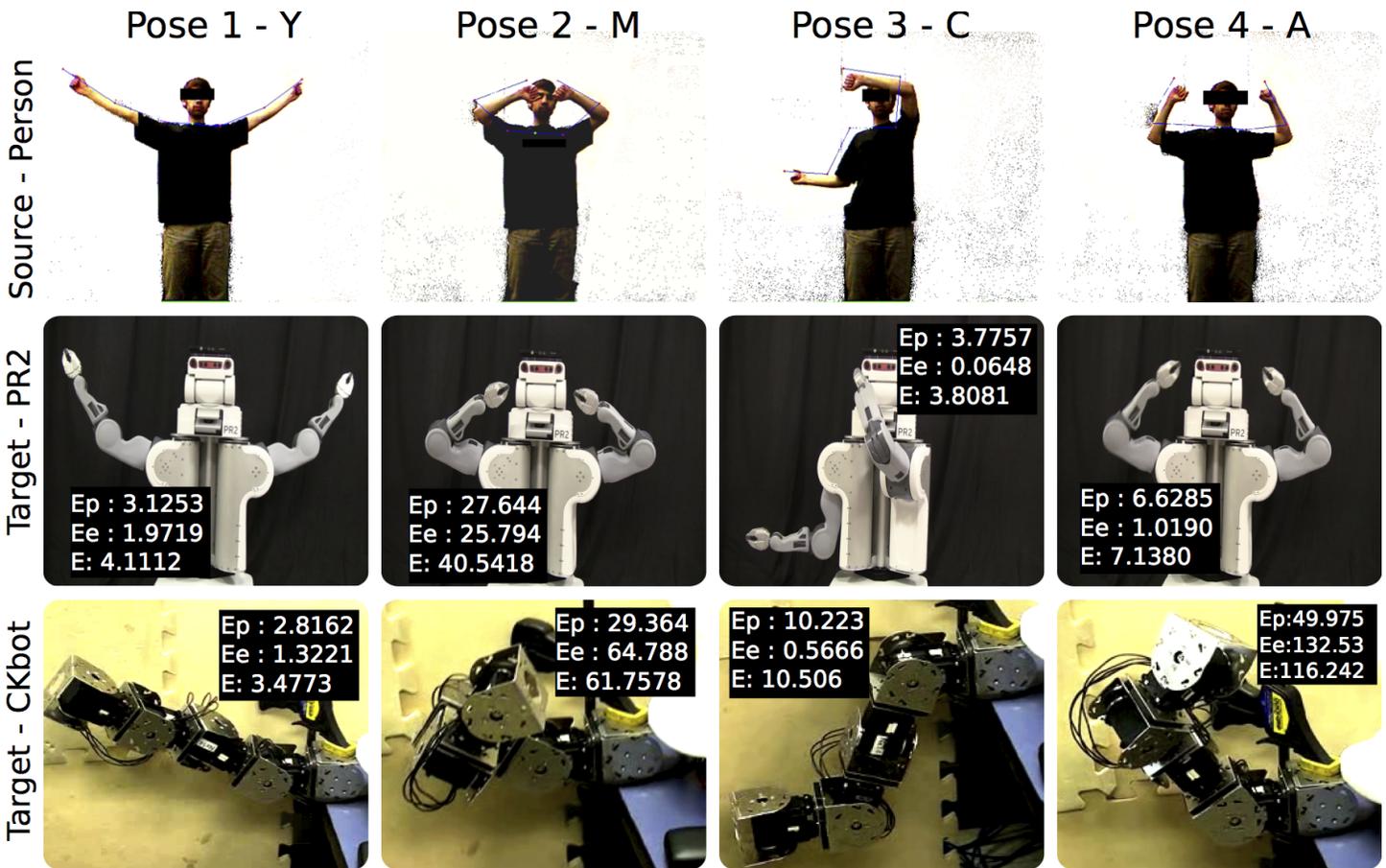


FIGURE 5: YMCA motion comparison. Metric values shown have been multiplied by 1E3 for ease of readability.

### 5.1 Retargeting from a Human Source Chain

In our system, motion is retargeted offline on a pose-by-pose basis. Our Kinect setup provides a vector of human joint positions in 3D at a rate of 30Hz. For our experiments, joints corresponding to the shoulder, elbow, and hand were used, as shown in Figure 6. This corresponds to 4 degrees of freedom of the human body: three in the shoulder and one in the elbow (movement of the hand relative the wrist is not captured). These points define the source chain for retargeting. To retarget a frame of motion capture data to a robot, the length of the human arm source chain is normalized to one, and the numerical optimization procedure described previously is used to find an optimal configuration for the normalized form of the robot arm chain model. This configuration is recorded and is also used as the initial condition for the optimization procedure in the next frame. Once all frames have been processed, the retargeted motion can be played back on the target robot.

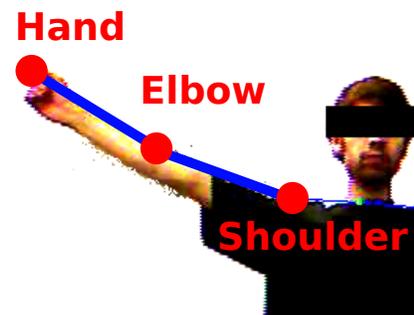


FIGURE 6: Human source chain: shoulder, elbow, and wrist joints are extracted from Kinect motion capture.

### 5.2 Retargeting to Robot Target Chains

Two robot platforms were used in our experiments. The first is the PR2 (Figure 7a), a humanoid robot with arms that resemble

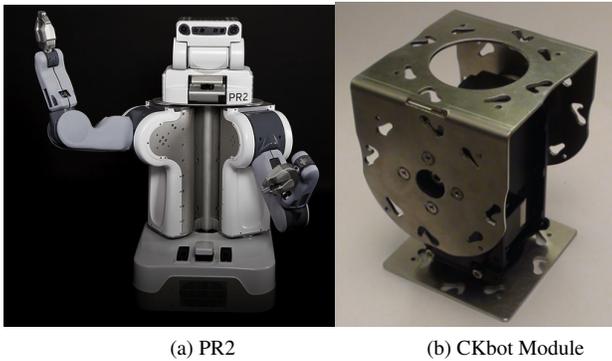


FIGURE 7: Robots used in experiments. Images are not shown to scale.

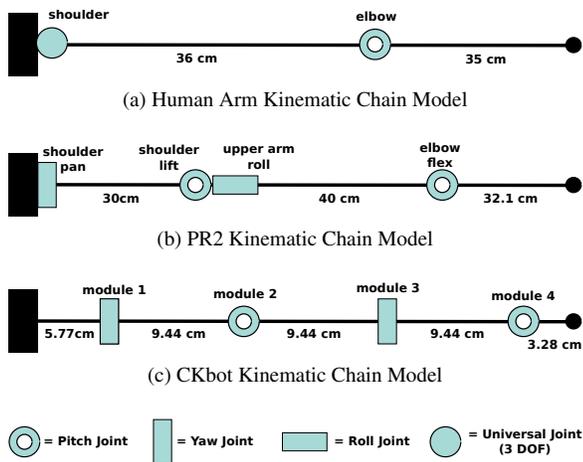


FIGURE 8: Arm Kinematic Chain Models

human arms [20]. Four degrees of freedom of the PR2 arm were used (Figure 8b). The second platform is the CKbot (Figure 7b), a chain-architecture modular robot designed at the University of Pennsylvania [21]. Each CKbot module includes a servomotor-actuated joint, to which a series of additional modules can be connected end-to-end to create complex structures. The freedom to easily create various kinematic chains makes the CKbot an excellent case study to demonstrate the strengths of our method. In our experiments, four CKbot modules were assembled into a 4-DOF arm. The kinematic structure consists of alternating pitch and yaw joints, each able to move 90-degrees in either direction. This arm was intentionally constructed to be very different from a human arm, in order to test the ability of the method to re-target between kinematically dissimilar chains. Figure 8 shows kinematic chain models for the PR2 arm, the CKbot arm, and a human arm.

### 5.3 Results

A “YMCA” gesture (Figure 5) was re-targeted to the PR2 and Ckbot arms. In this experiment, an  $\alpha$  value of 0.5 was used (discussed in Sec. 4.2). Only the right arm of the human is re-targeted to the CKbot arm. Table 1 provides average values of  $E_p$ ,  $E_e$ , and  $E$  over the four poses. Retargeting to the PR2 produces a convincing adaptation of the gesture. Retargeting to the CKbot arm produces a recognizable adaptation, though of lower quality than that produced for the PR2. Comparing the  $E$  (retargeting error) values, we see that the PR2 performs better on average than the CKbot arm.  $E_p$  (pose error) values are close between the two chains, but on average the PR2 performs better.  $E_e$  (end-effector error) values for the PR2 are much lower than the CKbot arm in the M, C, and A poses. This is intuitive: the PR2 arm is kinematically more similar to the human source, so our method is able to find solutions which closely match both the pose and the end-effector position of the source. The CKbot is articulated very differently than the source arm, resulting in a significant trade-off between end-effector distance and pose similarity (as discussed in Sec. 4.2). This trade-off makes the solutions much more sensitive to changes in  $\alpha$ , and results in optimal solutions with high  $E_e$  values.

	PR2	CKbot
$E_e$	7.20E-3	4.98E-2
$E_p$	1.03E-2	2.31E-2
$E$	1.39E-2	4.80E-2

TABLE 1: Average metric values for YMCA gesture

In frame 3 of the YMCA gesture, the PR2 is unable to closely match the pose of the human’s right arm because of a workspace limitation - the PR2 is unable to pitch its shoulder upwards more than twenty-five degrees. As a result the re-targeted pose for the right arm has high values of both  $E_p$  and  $E_e$ <sup>1</sup>. Faced with a target that cannot possibly match the source pose, the method finds a minimum-error solution which respects the target’s physical limitations.

## 6 Conclusion

This paper presents a method which can adapt poses to new robots with very little configuration by the user. We accomplish this by defining two dimensionless metrics for re-targeting: the

<sup>1</sup>Values listed in Figure 5 for the “C” pose of the YMCA gesture are for the PR2’s right arm. Left arm values are:  $E_p$ : 72.6016,  $E_e$ : 46.6676,  $E$ : 95.9354

pose error metric  $E_p$ , which is the integrated squared error between the normalized pose functions of the source and target, and the end-effector error metric  $E_e$ , which is the square of the distance between the end-effectors of the normalized forms of the source and target. The metrics may be applied to any source and target chains with revolute joints, even if they vary in link lengths or degrees of freedom.

This method has relevance in HRI as a means of adapting gestures from humans to robots, and gesture libraries between robots. Simulations and experiments show that it is effective in adapting poses across chains with different numbers of joints, and in adapting gestures from a human arm to two very different robots arms. Motion is retargeted on a frame-by-frame basis, with high enough fidelity that socially expressive gestures can be adapted from a human to a robot and between robots.

Opportunities for future work exist. The metrics presented could be expanded to include orientation, which may be important in some applications. Future work might also explore the application of this method in real-time, especially by using numerical differential kinematics (as in [12]) in place of numerical optimization to create an on-line controller.

## Acknowledgments

This work was sponsored by the National Science Foundation, and by the Princeton University department of Mechanical and Aerospace Engineering. We thank the UPenn Modlab and the USC Interaction Lab. We also thank Professor Maja Matarić (USC), Professor Mark Yim (UPenn), Terry Sun (UPenn), and Jean Mendez (UPenn), and Dylan Foster (USC).

## REFERENCES

- [1] Mead, R., and Salem, M., 2012. “Workshop on Speech and Gesture Production in Virtually and Physically Embodied Conversational Agents”. In International Conference on Multimodal Interaction, pp. 607–608.
- [2] Mead, R., Wade, E., Johnson, P., Clair, A. S., Chen, S., Matarić, M. J., and Member, S., 2010. “An Architecture for Rehabilitation Task Practice in Socially Assistive Human-Robot Interaction”. In Proceedings of The 19th IEEE International Symposium in Robot and Human Interactive Communication, pp. 404–409.
- [3] Matarić, M. J., and Pomplun, M., 1998. “Fixation behavior in observation and imitation of human movement.”. *Brain research. Cognitive brain research*, 7(2), Oct., pp. 191–202.
- [4] Gleicher, M., 1998. “Retargetting motion to new characters”. *Proc. of the 25th Ann. Conf. on Computer graphics and interactive techniques - SIGGRAPH '98*, pp. 33–42.
- [5] Hecker, C., Raabe, B., Enslow, R. W., DeWeese, J., Maynard, J., and van Prooijen, K., 2008. “Real-time motion retargeting to highly varied user-created morphologies”. *ACM Trns. on Graphics*, 27(3), Aug., p. 1.
- [6] Pollard, N., Hodgins, J., Riley, M., and Atkeson, C. “Adapting human motion for the control of a humanoid robot”. *Proc. 2002 IEEE Intl. Conf. on Robotics and Automation (Cat. No.02CH37292)*, 2, pp. 1390–1397.
- [7] Ude, A., Atkeson, C. G., and Riley, M., 2004. “Programming full-body movements for humanoid robots by observation”. *Robotics and Autonomous Systems*, 47(2-3), June, pp. 93–108.
- [8] Nakaoka, S., Nakazawa, a., Kanehiro, F., Kaneko, K., Morisawa, M., Hirukawa, H., and Ikeuchi, K., 2007. “Learning from Observation Paradigm: Leg Task Models for Enabling a Biped Humanoid Robot to Imitate Human Dances”. *The Intl. J. of Robotics Research*, 26(8), Aug., pp. 829–844.
- [9] Gielniak, M. J., and Thomaz, A. L., 2011. “Spatiotemporal correspondence as a metric for human-like robot motion”. In Proc. of the 6th Intl. Conf. on Human-robot interaction, HRI '11, ACM, pp. 77–84.
- [10] Gielniak, M. J., Liu, C. K., and Thomaz, A. L., 2010. “Secondary action in robot motion”. *19th Intl. Symposium in Robot and Human Interactive Communication*, Sept., pp. 310–315.
- [11] Dariush, B., Gienger, M., Jian, B., Goerick, C., and Fujimura, K., 2008. “Whole body humanoid control from human motion descriptors”. *2008 IEEE Intl. Conf. on Robotics and Automation*, May, pp. 2677–2684.
- [12] Dariush, B., Gienger, M., Arumbakkam, A., Goerick, C., and Fujimura, K., 2008. “Online and markerless motion retargeting with kinematic constraints”. *2008 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Sept., pp. 191–198.
- [13] Ott, C., Lee, D., and Nakamura, Y., 2008. “Motion capture based human motion recognition and imitation by direct marker control”. In 8th IEEE-RAS International Conference on Humanoid Robots, 2008, Ieee, pp. 399–405.
- [14] Boyd, S., and Vandenberghe, L., 2004. *Convex Optimization*. Cambridge University Press, Cambridge.
- [15] Ben-Tal, A., and Nemirovski, A., 2001. “Lectures on Modern Convex Optimization: Analysis, Algorithms and Engineering Applications”. In MPS-SIAM Series on Optimization.
- [16] MATLAB, 2011. `fmincon`. <http://www.mathworks.com/help/optim/ug/fmincon.html>.
- [17] OPENNI ORGANIZATION, 2010. *OpenNI User Guide*, November. Last viewed 19-01-2011 11:32. <http://www.openni.org/documentation>.
- [18] PRIMESENSE INC., 2010. *Prime Sensor NITE 1.3 Algorithms notes*. Last viewed 19-01-2011 15:34.
- [19] Kroon, D. J., 2011. `Kinect Matlab`. <http://www.mathworks.com/matlabcentral/fileexchange/30242->

kinect-matlab.

- [20] Cousins, S., 2010. “Ros on the pr2 [ros topics]”. *Robotics Automation Magazine, IEEE*, 17(3), sept., pp. 23–25.
- [21] Sastra, J., 2012. “Using modular reconfigurable robots for rapid development of dynamic locomotion experiments”. PhD thesis, University of Pennsylvania.