Autonomous Mapping between Motions and Labels

Junyun Tay[1,2], I-Ming Chen[1] and Manuela Veloso[2]

***Abstract*— A labeled motion library, in which robot motions are associated with semantic meanings, e.g., words, is useful for human-robot interaction, as a robot can use it to autonomously select motions to support its non-verbal communication. Manually assigning labels to new motions to a motion library is time consuming. However, a new motion may be similar to motions in the labeled motion library, and can be mapped to existing labels. We formally define motions, labels, and mappings between motions and labels. We use a NAO humanoid robot as a motivating example, though our approach is general for use on a humanoid robot with rotational joints. We explain how we generate motions and labels, define eight distance metrics to determine the similarity between motions, and use the nearest neighbor algorithm to determine the labels of a new motion. The distance metrics are varied across three axes - Euclidean versus Hausdorff, joint angles versus points of interest (postures), and mirrored versus non-mirrored. We evaluate the efficacy of these eight distance metrics, using precision, recall, and computational complexity.**

I. Introduction

Body motions such as gestures can be used by humanoid robots to communicate or reinforce what the robots are expressing. In order to autonomously add meaningful movement for a humanoid robot, motions must be labeled so as to select the right motions from the motion library. Maintaining the mappings between motions and labels is tedious and unfeasible when the motion library is large.

In this paper, we consider adding new motions into an existing labeled motion library, where each motion in the motion library has one or more labels. Without an autonomous way to map labels to new motions, all existing labels have to be examined manually to determine the mappings between labels and the new motion. We contribute an algorithm that autonomously determines mappings between a new motion and existing labels, by finding similar motions and using the labels of the similar motions as the labels for the new motion. We formally define motions for humanoid robots with rotational joints, the labels for the motions and the mappings between motions and labels. We investigate how to associate labels with a new motion by determining effective metrics to compute the similarity between two motions so as to use the labels of the most similar motion.

A similar motion has generally been defined as having similar joint angles or postures. We explore using joint angles and postures as measures to determine similarity. We also investigate two general distance metrics – Euclidean and Hausdorff distances. We introduce the concept of a mirrored motion, where a motion is symmetrical to another motion, e.g., where a motion involving the left hand or the right hand can be mapped to the same label. We incorporate all these approaches into eight distance metrics and compare the efficacy of each metric using precision and recall.

We use a NAO humanoid robot as a motivating example, though our approach is general for humanoid robots. We conducted experiments in Webots, a real-time simulator to determine the efficacy of the eight distance metrics. We explain how we create two motion libraries to compare a motion library with mirrored motions versus a library without. We also generate variants of the motions in each motion library to evaluate the distance metrics. We determine the mappings of existing labels to new motions using the eight distance metrics and the nearest neighbor algorithm. We determine the best distance metric based on precision, recall and their computational complexity.

II. Related Work

Tay and Veloso formally defined gestures and proposed how to animate speech with gestures autonomously by having a labeled motion library [1]. We build upon their work and explore how to expand the motion library when new motions are added.

Xia et al. contributed algorithms to enable humanoid robots to autonomously dance to the beats and emotions of music [2]. Xia et al. autonomously mapped emotional labels to motions by comparing different emotional static postures to the keyframes of a motion. However, only a single emotional label is assigned to each motion. We look at how we can map multiple labels to each motion.

Erdogan and Veloso analyzed the similarities between pairs of two-dimensional robot trajectories using the Euclidean distance between points and the Hausdorff metric [3]. Using the similarities derived, Erdogan and Veloso use a variant of agglomerative hierarchical clustering to determine groups of similar robot trajectories [3].

However, Erdogan and Veloso's method is used to find a cluster of trajectories that is assigned to only one group. Their method cannot be applied in a many-to-many relationship where a cluster of trajectories is assigned to one group and a subset of the trajectories assigned to another. For our task of assigning labels to a new motion, some of the labels can be assigned to other motions as well. Thus, a motion can be mapped to different labels and a label can be mapped to many motions, resulting in a many to many relationship.

[1]Junyun Tay and I-Ming Chen are with Nanyang Technological University, Singapore, junyunt@andrew.cmu.edu, michen@ntu.edu.sg

[3]Junyun Tay and Manuela Veloso are with Carnegie Mellon University, Pittsburgh PA, USA, junyunt@andrew.cmu.edu, veloso@cs.cmu.edu

Researchers have also looked at how to determine similarity between motions of humanoid robots and humans, mostly by comparing the joint angles [4], [5]. Huang et al. used a similarity function that compares the joint angles and velocities with a parameter that is adjusted to weigh the similarity between the spatial effect and temporal effect [6]. We consider both effects by varying the joint angles and velocities separately and both joint angles and velocities.

III. Formalization

In this section, we formally define motions for humanoid robots, starting with the definition of a keyframe (pose) and motion primitive. Each motion primitive is labeled with word(s) and has a duration defined either by using the fastest joint angular speeds or defined by the motion choreographer. The duration of the motion primitive can be varied using a multiplier. Interpolations between motion primitives (from the last keyframe of one motion to the first keyframe of the next motion) are also defined.

A. Robot Motions

We first begin by defining a robot:

Definition 3.1: A humanoid robot, R, has D actuated rotational joints with the corresponding joint limits and velocities, $\{(J_1, L_{1,\min}, L_{1,\max}, V_{1,\max}), \ldots, (J_D, L_{D,\min}, L_{D,\max}, V_{D,\max})\}$, $J_i \neq J_j$. For each joint J_d, the minimum and maximum angle of the joint is $L_{d,\min}$, $L_{d,\max}$ and the maximum velocity is $V_{d,\max}$. Let ζ be the D-dimensional configuration space of R.

1) Keyframe: A keyframe (static pose) stores the joints and corresponding angles at a particular time step. For the robot to perform a motion, several keyframes are stored at different times and interpolated to form a continuous motion.

Definition 3.2: A keyframe $k \in \zeta$ is a vector of D real numbers for each of the joint angles of R. A keyframe is $k_f = \{(J_1, \theta_1), \ldots, (J_n, \theta_n)\}, J_i \neq J_j$ and $n \leq D$. The joint index is J_d and the joint angle is θ_d. A keyframe is valid if the angles are within joint limits and the body parts of R do not collide. Let K_f be the set of all keyframes.

B. Motion Primitives (MPs)

Motions express an idea or meaning. To execute motions on a robot, the joints of the robot have to be actuated. A robot can only actuate its joints within the angular joint limits and speeds. A sequence of motion is made up of several motion primitives. A motion primitive is parameterized to allow the motion to be synchronized with the speech.

Definition 3.3: A **motion primitive** g is a tuple of $\mathtt{N}$ primitives and is parameterized with β. $g(\beta) = (\mathcal{M}_1, \ldots, \mathcal{M}_\mathtt{N})$ and $\mathtt{N} \in \mathbb{Z}^+$. The primitive $\mathcal{M}_n$ is a tuple of 2 keyframes, k_{n-1} and k_n, and the time to interpolate between these two keyframes, $t_{n-1,n}$, where $\mathcal{M}_n(\beta) = (k_{n-1}, \beta t_{n-1,n}, k_n)$. k_0, the first keyframe in $\mathcal{M}_1$, is the initial pose of the robot, R, which contains all the joint angles for D degrees of freedom. Let $M = \bigcup g$ be the set of all motion primitives in the motion library.

The motion primitive is parameterized with β, where $\beta \in \mathbb{R}$ and $\beta \geq 1$. β is used as a multiplying factor that allows the execution of the motion primitive to be slowed by a multiplier. To create similar motion trajectories, we use β to vary the motion.

To interpolate between two keyframes or motion primitives, we use linear interpolation. We assume the interpolations between keyframes are generated by a motion planner and are collision-free and are within physical capabilities (joint angular and velocity limits). The time to interpolate between two keyframes, k_n and k_{n+1}, is computed by the interpolation time computation function $T : K_f \times K_f \rightarrow \mathbb{R}^+$, i.e., $t_{n,n+1} = T(k_n, k_{n+1})$. $t_{n,n+1}$ specifies the minimum duration required to interpolate from the respective joint angles in k_n to the angles defined in k_{n+1} or is pre-defined by the motion choreographer.

A label represents the meaning expressed by the motion. Each label is mapped to one or more motions. Each motion is mapped to one or more labels. Hence, mappings between motions and labels is a many-to-many function.

Definition 3.4: A label, l, consists one or more words. Let L be the set of all labels.

Definition 3.5: Let M be a library of motion primitives and L be a set of labels for the meanings. The function $\mathbb{X} : M \times L \rightarrow \{0, 1\}$ determines the mapping between the motions and the labels, i.e., $\mathbb{X}(m, l) = 1$ if the motion $m \in M$ is mapped to the label $l \in L$.

This paper aims to find the function $\mathbb{X}^+ : M^+ \times L \rightarrow \{0, 1\}$, so as to map labels L to a new motion m^+, where $M^+ = M \cup \{m^+\}$, given the existing mapping function $\mathbb{X}$, and given that m^+ is similar to an existing motion in the motion library M.

IV. Technical Approach

In this section, we present how we create and expand the motion library, describe eight distance metrics to compare different variants of motions, and how we map existing labels to a new motion that is added into the motion library.

A. Creating the Motion Library

Seventy words were taken from a list of words that toddlers should know [7], the Dolch word list, "a list of frequently used English words compiled by Edward William Dolch" [8] and Paul Ekman's six basic emotions as labels. We trained a group of high school students to create motions using the NAO humanoid robot and Choregraphe [9], a software to create keyframe motions. The students created two to three motions for each label.

Each motion is defined as a motion primitive, where some of the interpolation times are defined by the choreographer so that each motion is stable for execution. The rest of the interpolation times are calculated based on 80% of the maximum joint velocities. The motions for Paul Ekman's six basic motions were modified from the motions available at http://hcm-lab.de/projects/shr [10]. In total, there are 170 motions.

Each motion in the motion library is tested in the simulator, Webots 7 [11], to ensure that the NAO humanoid robot is stable after executing the motion. Webots 7 [11] is a real-time simulator that stimulates the dynamics of the NAO humanoid robot. If the motion is unstable, the interpolation times are adjusted until the motion is stable.

After ensuring the stability of each motion, a video of the NAO humanoid robot executing each motion is shown to another group of students and they are asked to provide labels for each motion. Hence, more labels are added, resulting in 170 motions and 322 labels.

B. Expanding the Motion Library

We term the motions in the motion library, Initial, which are motions with no modifications. Some of the motions in Initial are mirrored motions. Since our motion library consisted of 170 motions, we expand Initial to 1700 motions by varying the joint angles and/or interpolation times.

1) Varying Interpolation Time and Joint Angles: To create similar motions, we created variants of Initial in the motion library by varying the following features and assume each variant of a motion, g_n share the same labels assigned to g_n in Initial:

- ModJoints: We only modify the joint angles of each motion where each joint angle for each keyframe in the motion primitive can be modified with a 50% probability. If the joint angle is modified, the joint angle will be changed within a range of -5 to 5 degrees, so $\theta_d = \theta_d + \Delta$ and $\Delta \in \{-5^o, -4^o, -3^o, \ldots, 5^o\}$.
- ModTime: We only vary the interpolation times by changing β, $\beta \in \{1.25, 1.5, 1.75, 2\}$.
- ModJointsAndTime: We change both joint angles and interpolation time of each motion by combining the first and second features. We use the motions that are modified in ModJoints and modify the interpolation time by $\beta \in \{1.25, 1.5, 1.75, 2\}$.

We assume that the labels of Initial are still applicable to ModJoints, ModTime and ModJointsAndTime. In other words, the variants of Initial share the same labels as Initial.

2) Creating Mirrored Motions: The NAO humanoid robot has 25 actuated rotational joints. Figure 1 shows the positions of all the joints in the NAO humanoid robot. Although Fig. 1 show a total of 26 joints, only 25 joints are actuated as the LHipYawPitch and RHipYawPitch "share the same motor so they move simultaneously and symmetrically" and in the case of "conflicting orders, LHipYawPitch always takes the priority" [13].

Some motions are a mirror image of another motion in the library; we term them *mirrored motions*. The meanings of a motion can be similar when a motion is a mirror image of another motion. For example, waving with the left hand and waving with the right hand can express the same label – "wave". Also, kicking with the left leg and kicking with the right leg can also express the label – "kick". However, waving with the left hand can be labeled with the phrase 'wave with left hand' and waving with the right hand can also be labeled with the phrase 'wave with right hand'. Therefore, though they share most of their labels (and meanings), they can also be mapped to different labels.

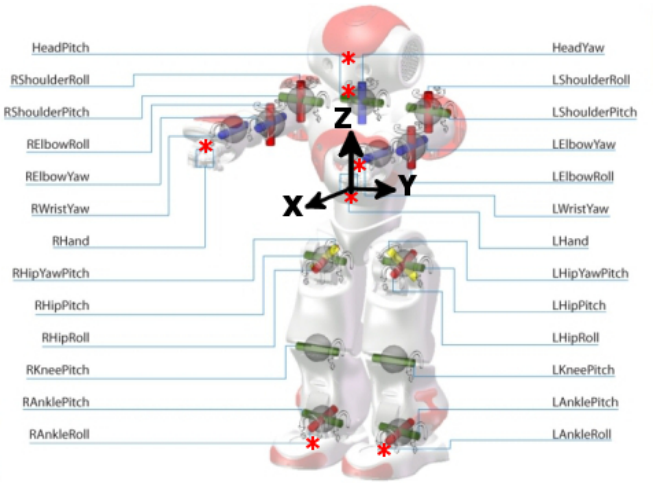


Fig. 1: Joints, POIs and coordinate frame of the NAO robot. Edited image from [12]

Paired Joints		Corresponding Mirrored Joints	
HeadYaw	HeadPitch	-HeadYaw	HeadPitch
LShoulderPitch	RShoulderPitch	RShoulderPitch	LShoulderPitch
LShoulderRoll	RShoulderRoll	-RShoulderRoll	-LShoulderRoll
LElbowYaw	RElbowYaw	-RElbowYaw	-LElbowYaw
LElbowRoll	RElbowRoll	-RElbowRoll	-LElbowRoll
LWristYaw	RWristYaw	-RWristYaw	-LWristYaw
LHand	RHand	RHand	LHand
LHipRoll	RHipRoll	-RHipRoll	-LHipRoll
LHipPitch	RHipPitch	RHipPitch	LHipPitch
LKneePitch	RKneePitch	RKneePitch	LKneePitch
LAnklePitch	RAnklePitch	RAnklePitch	LAnklePitch
LAnkleRoll	RAnkleRoll	-RAnkleRoll	-LAnkleRoll
LHipYawPitch		LHipYawPitch	

TABLE I: Paired Joints and Corresponding Mirrored Joints

We compute a mirrored motion by looking at pairs of joints that are symmetrical to each other by the Z axis in Fig. 1 using the function mirror, where $\text{mirror}(J_{original}) = J_{mirrored}$. Table I shows the list of 25 joints and the corresponding mirrored joints. For example, for the joint angle HeadYaw of the original motion, $HeadYaw_{\text{original}}$, the joint angle HeadYaw for the mirrored motion will be negative, hence to find $HeadYaw_{\text{mirrored}}$, we use the function $\text{mirror}(HeadYaw_{\text{original}}) = -HeadYaw_{\text{original}}$.

C. Similarities of motion trajectories

Besides joint angles, we also consider the differences in the three-dimensional positions of the joints with respect to the robot's torso as the joint differences may not reflect the differences in posture. Hence, we compute the three-dimensional (3D) position of each joint of the robot and termed each position as a point of interest (POI). Besides each joint, the points of interest (POIs) also include seven red asterisks (*) (Fig. 1). We add these seven POIs as the 3D positions of these seven POIs vary with joint angles changes in the head, wrists and ankle joints, whereas the 3D positions of the head, wrists and ankle joints are invariant to

joint angle changes. E.g., the 3D position of the HeadYaw joint remain unchanged when HeadYaw's joint angle change. Besides using Euclidean distance, Erdogan and Veloso also chose "the Hausdorff metric for its generality and efficiency" [3]. Eight distance metrics are varied across three axes – Euclidean versus Hausdorff, mirrored versus non mirrored and joint angles versus POIs:

1) EuclideanJoint: We compute the average absolute joint difference between the same joint for two different motions for each time step. If a motion g_1 is longer in duration than the other motion g_2, we use the joint angles at the last time step of g_2 to compare with the rest of the joint angles of g_1, and vice versa. Let the duration of g_1 be t_1 and the duration of g_2 be t_2. We determine the average joint difference:

$$\text{EuclideanJoint}(g_1,g_2) = \frac{\sum_{d=1}^{25}\sum_{t=1}^{\max(t_1,t_2)}|J_{d_t}^{g_1} - J_{d_t}^{g_2}|)}{\max(t_1,t_2)}$$

2) EuclideanMirrorJoint: We compute the difference in joint angles for motion g_1 and a mirrored motion of another motion g_2 using the function mirror which calls mirror(J_i) on each joint J_i of g_2 in each timestep. We also compute the difference in joint angles for g_1 and g_2 and use the smaller average joint difference:

$$\begin{aligned}&\text{EuclideanMirrorJoint}(g_1,g_2)\\ &\quad=\min(\text{EuclideanJoint}(g_1,g_2),\\ &\qquad\text{EuclideanJoint}(g_1,\text{mirror}(g_2)))\end{aligned}$$

3) EuclideanPOI: We compute the average absolute Euclidean distance of the 3D position of the same POI for two different motions for each time step. If a motion g_1 is longer in duration than the other motion g_2, we use the 3D position of the POI at the last time step of g_2 to compare with the rest of the 3D position of the same POI of g_1, and vice versa. Let the duration of g_1 be t_1 and the duration of g_2 be t_2. We determine the average Euclidean POI difference:

$$\text{EuclideanPOI}(g_1,g_2) = \frac{\sum_{p=1}^{32}\sum_{t=1}^{\max(t_1,t_2)}|\text{POI}_{p_t}^{g_1} - \text{POI}_{p_t}^{g_2}|}{\max(t_1,t_2)}$$

4) EuclideanMirrorPOI: We compute the average absolute Euclidean distance of the three dimensional position of the same POI for two different motions for each time step. We also compute the average absolute Euclidean distance of the first motion to the mirrored motion of the second motion and take the minimum:

$$\begin{aligned}&\text{EuclideanMirrorPOI}(g_1,g_2) =\\ &\quad\min(\text{EuclideanPOI}(g_1,g_2),\\ &\qquad\text{EuclideanPOI}(g_1,\text{mirror}(g_2)))\end{aligned}$$

5) HausdorffJoint: Instead of only Euclidean distances between joints or POIs, we use the Hausdorff metric, where $d(J_{g_1}, J_{g_2})$ is the Euclidean distance between two joints:

$$\begin{aligned}&\text{HausdorffJoint}(g_1,g_2) =\\ &\quad\max(\max_{J_{g_1}\in g_1}\min_{J_{g_2}\in g_2} d(J_{g_1},J_{g_2}),\\ &\qquad\max_{J_{g_2}\in g_2}\min_{J_{g_1}\in g_1} d(J_{g_1},J_{g_2}))\end{aligned}$$

6) HausdorffMirrorJoint: We use HausdorffJoint to find the minimum of two Hausdorff measures – joint angles for g_1 and g_2 and joint angles for g_1 and mirror(g_2):

$$\begin{aligned}&\text{HausdorffMirrorJoint}(g_1,g_2) =\\ &\quad\min(\text{HausdorffJoint}(g_1,g_2),\\ &\qquad\text{HausdorffJoint}(g_1,\text{mirror}(g_2)))\end{aligned}$$

7) HausdorffPOI: Instead of joint angles, we look at Hausdorff measures for POIs and $s(\text{POI}_{g_1}, \text{POI}_{g_2})$ is the Euclidean distance between two POIs of g_1 and g_2:

$$\begin{aligned}&\text{Hausdorff\_POI}(g_1,g_2) =\\ &\quad\max(\max_{\text{POI}_{g_1}\in g_1}\min_{\text{POI}_{g_2}\in g_2} s(\text{POI}_{g_1},\text{POI}_{g_2}),\\ &\qquad\max_{\text{POI}_{g_2}\in g_2}\min_{\text{POI}_{g_1}\in g_1} s(\text{POI}_{g_1},\text{POI}_{g_2}))\end{aligned}$$

8) HausdorffMirrorPOI: We use HausdorffPOI to compute the minimum of two Hausdorff measures – POIs for motion g_1 and g_2 and POIs for g_1 and mirror(g_2):

$$\begin{aligned}&\text{HausdorffMirrorPOI}(g_1,g_2) =\\ &\quad\min(\text{HausdorffPOI}(g_1,g_2),\\ &\qquad\text{HausdorffPOI}(g_1,\text{mirror}(g_2)))\end{aligned}$$

D. Adding a new motion to the motion library

We use the nearest neighbor algorithm to select the closest motion to the new motion using the output of a distance metric and map its labels to the new motion:

First, given a new motion m^+, and the existing motion library M, using $D(m^+, m)$, one of the distance metrics, e.g., $EuclideanJoint$, we find:

$$m^* = \text{argmin}_{m\in M} D(m^+, m)$$

Second, we create an updated motion library $M^+ = M \cup \{m^+\}$. Third, a new motion m^+ is mapped to m^*'s labels and use the updated mapping function $\mathbb{X}^+$:

$$\mathbb{X}^+(m,l) = \begin{cases}\mathbb{X}(m,l) & \text{if } m^+ \neq m\\ \mathbb{X}(m^*,l) & \text{otherwise}\end{cases}$$

Thus, the new motion and its corresponding labels are represented in the updated motion library M^+ and the updated mapping function $\mathbb{X}^+$.

V. EXPERIMENTS

In this section, we describe our experiments to evaluate the eight distance metrics and the nearest neighbor algorithm that autonomously maps motions to labels.

A. Experimental Setup

We compare the different distance metrics to determine similarities of motion trajectories. We use the motions from an existing motion library used by a NAO to animate stories – Original – and create another motion library – NoMirrored – by removing mirrored motions from Original. These two motion libraries enable us to understand the efficacy of including the function mirror in the distance metrics:

- Original: Original has a set of 170 motions and 322 associated labels.
- NoMirrored: NoMirrored has a set of 126 motions and 265 associated labels.

Next, we create variants described in Section IV-B.1 – Initial, ModJoints, ModTime and ModJointsAndTime for each of the 2 motion libraries – Original and NoMirrored.

B. Analysis of Distance Metrics

To evaluate the eight distance metrics, we used $\text{Precision} = \frac{\text{true positives}}{(\text{true positives}+\text{false positives})}$ and $\text{Recall} = \frac{\text{true positives}}{(\text{true positives}+\text{false negatives})}$ to measure the performance of assigning labels to motions. The term positive means the motion is assigned a label and negative means the motion is not assigned a label. The term true means the label assigned is right and false means the label assigned is wrong. We term true positives TP, false positives FP, false negatives FN and each term is indexed by v – the index of the label. The equation to compute the precision is $\sum_{v=1}^{|L|} \text{TP}_v / (\sum_{v=1}^{|L|} \text{TP}_v + \sum_{v=1}^{|L|} \text{FP}_v)$ and the equation for recall is $\sum_{v=1}^{|L|} \text{TP}_v / (\sum_{v=1}^{|L|} \text{TP}_v + \sum_{v=1}^{|L|} \text{FN}_v)$, where $|L|$ is the number of labels in the library.

We perform 10-fold cross validation, where the motions are randomly divided into 10 folds and we iteratively use 1 fold as test data and the rest as training data. Next, we determine the labels of each motion in the test data using a distance metric and the nearest neighbor algorithm. We perform the cross validation 10 times for each distance metric, find the precision and recall for each variant of motions in each motion library and summarize the results with a mean and standard deviation in Fig. 2 and Fig. 3.

The precision and recall for the Initial motions is low as compared to other variants of motions, e.g., ModTime, which is expected as the Initial motions do not have many similar motions. Hence, the nearest neighbor algorithm is unable to find an exact match of the labels for the new motion. In contrast, when the library is expanded with ModTime for example, the nearest neighbor algorithm returns a similar motion with the exact labels, and hence the precision and recall is higher.

For the Original motion library, distance metrics that include the function mirror perform worse than measures that do not – although most labels of the mirrored motions are the same as the labels of the Initial motions, some of them are different as they include the word, "right" instead of "left" or vice versa. Since there are no mirrored motions in the NoMirrored motion library, the precision and recall for the NoMirrored motion library are similar for metrics with or without the function mirror. This finding supports our explanation of why distance metrics that include the function mirror perform worse for the Original motion library than the NoMirrored motion library.

The distance metrics that involve Euclidean distances perform as well as the distance metrics that involve Hausdorff distances, however Hausdorff distances are computationally more expensive and runs in $O(t^2)$, whereas Euclidean distances run in $O(t)$, where t is the number of time steps of the longer motion. The distance metrics that involve the joints perform as well as distance metrics that involve the POIs. However, distance metrics that involve the POIs use more computations (absolute difference between a pair of 3D points) than the distance metrics that involve the joints as we take the absolute difference between each pair of joint angles. Hence, EuclideanJoint is the best distance metric for motions such as ModJoints, ModTime and ModJointsAndTime in terms of precision, recall and computational complexity.

VI. Conclusion

We formally defined motions, labels, and mappings between motions and labels. We also described eight distance metrics to determine the similarities of motions. We created two motion libraries and explained how we created variants of the motions to conduct experiments. We found mappings of existing labels to new motions using the eight distance metrics and the nearest neighbor algorithm. We presented the efficacy of each distance metric using precision and recall. We found that EuclideanJoint is the best distance metric in terms of precision, recall and computational complexity.

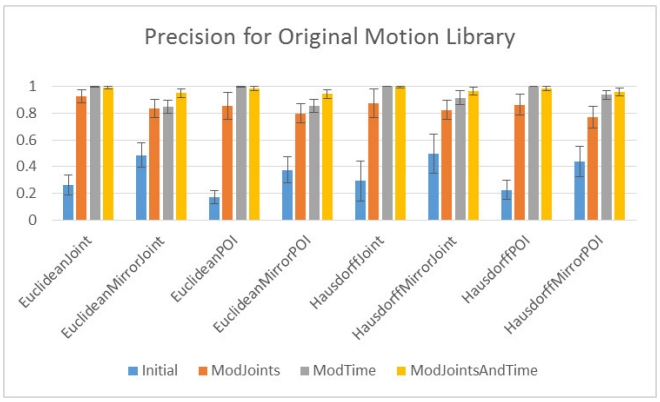
We observe that distance metrics with the mirror function have a lower precision and recall. We hypothesize that precision and recall can be increased by looking through the labels, and replacing the words associated with the mirrored motion, e.g., changing "left" to "right", but this approach requires a dictionary of such pairs of words.

Other distance metrics such as dynamic time warping, longest common subsequence, etc., have also been evaluated for motion trajectories [14] and time series data [15]. These distance metrics can also be evaluated to compare their performance in precision, recall and computational complexity.

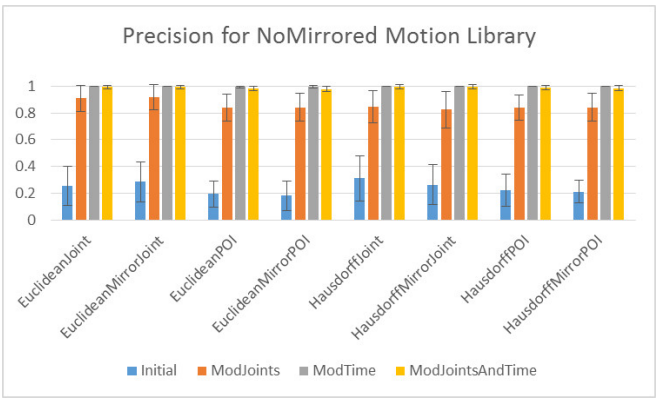
Our work does not consider the scenario that a new unseen motion is added to the library. As future work, we can investigate thresholds to determine if there are similar motions, so that if the similarity of a new motion to existing motions falls below the threshold, no labels can be mapped to the new motion. We can also assign a confidence value for each motion-label pair so as to determine if the label is applicable by learning from feedback of the audience when the robot executes the motion based on the label assigned. We currently assume that there are no new label(s) assigned to a new motion. Moreover, similar label(s) in semantic meanings to the new label(s) are determined so as to associate the new label(s) to the motions of the similar label(s).

Acknowledgments

This work was supported by the Singapore Millennium Foundation Research Grant, and by the National Science

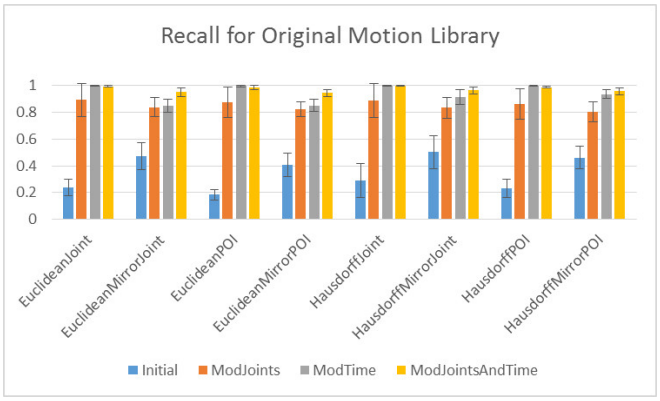


(a) Precision for Original Motions

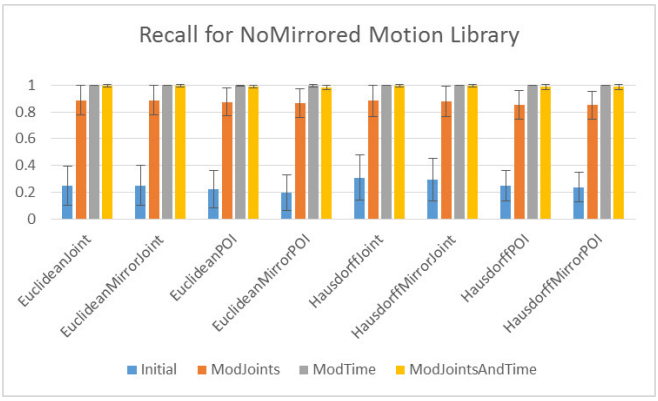


(b) Precision for NoMirrored Motions

Fig. 2: Precision for 2 motion libraries



(a) Recall for Original Motions



(b) Recall for NoMirrored Motions

Fig. 3: Recall for 2 motion libraries

Foundation under awards IIS-1012733 and IIS-1218932, and the A*STAR Computational Resource Centre through the use of its high performance computing facilities. Junyun Tay is part of the NTU-CMU Dual PhD Programme in Engineering (Robotics) which is funded by the Economic Development Board of Singapore. We thank Somchaya Liemhetcharat for his feedback and comments. The views and conclusions contained herein are those of the authors only.

REFERENCES

[1] J. Tay and M. Veloso, "Modeling and composing gestures for human-robot interaction," in *International Symposium on Robot and Human Interaction Communication (RO-MAN)*, 2012, pp. 107–112.

[2] G. Xia, J. Tay, R. Dannenberg, and M. Veloso, "Autonomous robot dancing driven by beats and emotions of music," in *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, vol. 1, 2012, pp. 205–212.

[3] C. Erdogan and M. Veloso, "Action selection via learning behavior patterns in multi-robot domains," in *International Joint Conference on Artificial Intelligence. (IJCAI)*, 2011, pp. 192–197.

[4] X. Zhao, Q. Huang, Z. Peng, and K. Li, "Kinematics mapping and similarity evaluation of humanoid motion based on human motion capture," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1, Sept 2004, pp. 840–845 vol.1.

[5] P. Sousa, J. L. Oliveira, L. P. Reis, and F. Gouyon, *Progress in Artificial Intelligence: 15th Portuguese Conference on Artificial Intelligence, EPIA 2011, Lisbon, Portugal, October 10-13, 2011. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, ch. Humanized Robot Dancing: Humanoid Motion Retargeting Based in a Metrical Representation of Human Dance Styles, pp. 392–406.

[6] Q. Huang, Z. Yu, W. Zhang, W. Xu, and X. Chen, "Design and similarity evaluation on humanoid motion based on human motion capture," *Robotica*, vol. 28, no. 5, pp. 737–745, Sept. 2010.

[7] Laura Mize, "Dolch word list," 2008. [Online]. Available: http://teachmetotalk.com/2008/02/12/first-100-words-advancing-your-toddlers-vocabulary-with-words-and-signs/

[8] Wikipedia, "First 100 words-advancing your toddler's vocabulary with words and signs," 2015. [Online]. Available: http://en.wikipedia.org/wiki/Dolch_word_list

[9] Aldebaran Robotics, "NAO software," 2014. [Online]. Available: http://doc.aldebaran.com/1-14/software/choregraphe/index.html

[10] M. Haring, N. Bee, and E. Andre, "Creation and evaluation of emotion expression with body movement, sound and eye color for humanoid robots," in *Int. Symp. on Robots and Human Interact. Comm. (RO-MAN)*, July 2011, pp. 204–209.

[11] Webots, "http://www.cyberbotics.com," 2014. [Online]. Available: http://www.cyberbotics.com

[12] Aldebaran Robotics, "NAO actuator and sensor list," 2014. [Online]. Available: http://doc.aldebaran.com/2-1/family/nao_dcm/actuator_sensor_names.html

[13] Aldebaran Robotics, "Effector and chain definitions," 2014. [Online]. Available: http://doc.aldebaran.com/2-1/family/robots/bodyparts.html

[14] O. Cigdem, T. D. Laet, and J. D. Schutter, "Classical and subsequence dynamic time warping for recognition of rigid body motion trajectories," in *Control Conference (ASCC), 2013 9th Asian*, June 2013, pp. 1–6.

[15] M. D. Morse and J. M. Patel, "An efficient and accurate method for evaluating time series similarity," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 2007, pp. 569–580.