JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 18, No. 1, March 2024, pp. 47-56

Real-Time Retargeting of Human Poses from Monocular Images and Videos to the NAO Robot

Oscar Burga, Jonathan Villegas, and Willy Ugarte*
Department of Computer Science, Universidad Peruana de Ciencias Aplicadas, Lima, Peru
u20181a324@upc.edu.pe, u20181c045@upc.edu.pe, willy.ugarte@upc.pe

Abstract
To this point, there has been extensive research investigating human-robot motion retargeting, but the vast majority of existing methods rely on sensors or multiple cameras to detect human poses and movements, while many other methods are not suitable for usage on real-time scenarios. The current paper presents an integrated solution for performing real-time human-to-robot pose retargeting utilizing only regular monocular images and video as input data. We use deep learning models to perform three-dimensional human pose estimation on the monocular images and video, after which we calculate a set of joint angles that the robot must utilize to reproduce the detected human pose as accurately as possible. We evaluate our solution on Softbank's NAO robot and show that it is possible to reproduce promising approximations and imitations of human motions and poses on the NAO robot, although it is subject to the limitations imposed by the robot's degrees of freedom, joint constraints, and movement speed limitations.

Category: Real-Time Systems

Keywords: Human pose estimation; Humanoid robot; Motion retargeting; Geometry; Vectors

I. INTRODUCTION

Human-robot motion retargeting has been established as an important part of the learning-from-demonstration paradigm in the robotics field, where it is often applied to train or develop motor function in humanoid robots. However, most existing solutions based on motion-retargeting rely on the usage of either single/multiple cameras observing the human demonstrator [1] or sensors to detect the source human pose [2].

Moreover, many of the existing solutions are not fit for usage in real-time scenarios due to excessive computational resource demands and other constraints.

Motion tracking and retargeting has applications in multiple fields, with the most common being in the robotics field for robot motion [3, 4]. Other applications in recent years have appeared in the fields of computer animation, such as for generating realistic human-like motion on different bodies. Motion tracking and pose estimation has also seen some applications, including in monitoring motor function development in children [5].

Similarly, these technologies have been applied in the field of sports for monitoring the performance and training of athletes [6] and for monitoring and preventing injuries in athletes [7]. Designing an appropriate solution for human motion tracking and retargeting requires solving many challenges from the computational perspective, with even more challenges to be solved if not using sensors, as is the focus of the current work. Not having access to sensors or multiple cameras means that it is necessary to

Open Access http://dx.doi.org/10.5626/JCSE.2024.18.1.47 http://jcse.kiise.org

Received 09 March 2023; **Accepted** 14 March 2024
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

extract meaningful and accurate three-dimensional (3D) information about human poses from two-dimensional RGB images.

In [8], the authors explain that this is itself a difficult problem due to depth ambiguities, inconsistent metrics (e.g., the need to estimate real-world measures such as millimeters instead of using image pixels), and the non-linear nature of human dynamics and motions. Moreover, the extracted 3D information has to then be further processed to calculate relevant data for the robot's motion, such as estimating joint angles.

The contributions of our present work are as follows:

- We trained a deep learning model to perform 3D human pose estimation from the two-dimensional image and video data.
- We developed our own inverse kinematics algorithm to obtain the corresponding joint angles to utilize for the robot.
- We developed an interface to stream the obtained joint angles to a physical NAO robot, which will essentially replicate the movements of a human demonstrator from either an image, a video, or a camera's video feed.

In Section II, we explore and discuss similar solutions to our problem. In Section III, we explain the main concept involved in developing our proposal, such as the chosen deep learning model, the inverse kinetics method, and the interface to the NAO robot. In Section IV, we detail all the experiments that we carried out and their results to prove the feasibility of our proposal. Finally, in Section V, we describe the main conclusions of this work and future directions.

II. RELATED WORKS

Several previous works have focused on motion retargeting from a human to a humanoid robot.

A. Solutions that Utilize Sensors

Some of these works utilize sensors to first obtain the pose of the human and then apply diverse techniques for retargeting.

One of the most well-known approaches for humanoid motion retargeting was performed in [2]. They developed a solution using one Kinect sensor and two strength sensors to obtain the movements performed by a person, after which a Gaussian process latent variable model, dynamic time warping (DTW), and variational hetero-scedastic Gaussian process regression algorithm were applied to perform the retargeting from a human pose to the joints of a NAO robot.

Another interesting approach is that which was developed in [9]. This approach involved using a Kinect sensor to obtain RGBD images of a human performing a pose and then introducing those images into their parametric model called HUMROB, which has the shape of the target skeleton but keeps the dimension of the body as similar as possible to those of the corresponding human. The joint's angles are obtained directly from the final model and then are sent to the humanoid robot.

Another notable approach was developed in [10]. In that study, the skeleton extraction was performed by a Kinect sensor, and the retargeting was manually done using linear algebra to calculate the angles in the robot coordinate space. The main focus of that work was on allowing the robot to perform complex motion sequences without losing balance. To that end, they applied clusterization and optimization to calculate the ideal sequence of angles and achieve a smoother motion without losing the original motion properties.

Our approach herein differs from these previous solutions because we are not using sensors to obtain the human position, as we instead focus uniquely on monocular images and videos as well as human pose estimation to obtain the pose. Our method for retargeting utilizes analytic geometry and linear algebra, and it therefore shares some similarities with [10] according to the field of study, but not for the specific approach.

B. Solutions that Utilize Human Pose Estimation

Meanwhile, there are also approaches that involve applying human pose estimation combined with other techniques.

One remarkable approach was developed in [11]. Their work involved the use of 3D pose estimation to obtain the pose of a person. In particular, two poses of the same person were obtained from different points of view, which were then fused and used to obtain the angles, which were finally directly transferred to the robot that has the same skeletal structure as the human.

Another approach developed in [12] involved the construction of a whole vision-based system that included a human pose estimation model and a series of analytical procedures to retarget the angles into a single armed robot.

The novelty of our approach relies in using one camera point that only uses one camera point of view for pose calculation, and that we present a slightly different solution for the arm angle calculation and also include the head movement.

III. RETARGETING HUMAN POSES TO THE NAO ROBOT

Three-dimensional motion tracking and retargeting from monocular images and videos is a challenging

problem due to the need to extract 3D information from 2D data. Managing multiple degrees of freedom for limbs and joints, and achieving smoothness between individual movements and maintaining good computational performance are some of the specific difficulties that have to be dealt with when performing such a task.

A. Preliminary Concepts

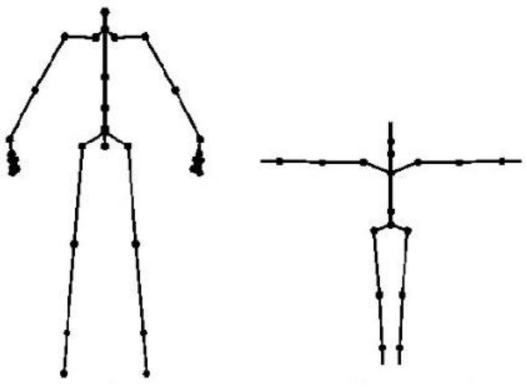
Next, we present some of the common key concepts related to motion retargeting from monocular image and video data.

1) Human Pose Estimation

According to [13], the task of human pose estimation aims to estimate the configuration of human body parts from some given input data, which is typically in the form of either images or videos. Human pose estimation allows for the extraction of geometric information pertaining to the analyzed human body, which can then be used for a wide variety of applications and purposes, as is the case with human-robot motion retargeting.

Human pose estimation methods tend to be categorized by authors, such as [9], into two main categories: marker-based methods and markerless methods. The former requires the person to wear sensors or some kind of hardware attached to keypoints on their body to obtain precise marker positions, while the latter only utilizes external sensors (most commonly a single camera) and often requires the use of deep learning models and algorithms to actually extract the keypoints data from the body.

Moreover, human pose estimation may also be performed in three dimensions. That is, the 3D pose of a person can be estimated from a 2D image or video. Popular methods for this often begin by utilizing 2D pose estimation, to obtain the given keypoints in the image, and then utilize a deep learning model or some method to "lift" the keypoints from the image and generate the data for the third dimension.



(a) (b)

Fig. 1. Example of retargeting of human joint structure: (a) source skeleton and (b) target skeleton.

2) Human-Robot Motion Retargeting

Motion retargeting consists of the transformation of a posture or motion from one hierarchical structure of joints to another target structure. In the specific context of the present work, human-robot motion retargeting is explained by [14] as the process of a robot mimicking human-like motion as accurately as possible (Fig. 1). In [9], the authors explain two of the main difficulties associated with motion retargeting: geometric and topologic differences.

Geometric differences: Differences in the size and length of the different body parts or limbs of the human and target robot. It is typically not too difficult to handle geometric differences on their own when performing motion retargeting, as long as the joint structure (topology) remains similar or equal between the source and target bodies. A common approach in such cases is to employ inverse kinematic solvers or similar algorithms to estimate the angles for each joint from their given location.

Topologic differences: Refers to the differences in hierarchical joint structure between the source human and the target robot; i.e., the robot is not of humanoid structure. Such cases are much more difficult to handle as no proper mapping exists by default, so it has to be generated or learned through different means. In the context of this work, we are dealing with very little topologic differences due to the usage of the NAO robot, but it is still important to keep them in mind.

B. Method

To perform motion retargeting of human poses from monocular images and videos to the NAO robot, we propose using an off-the-shelf 3D pose estimator to extract the 3D pose data from the input image, video, or camera feed. For the latter cases, pose smoothing is also performed across frames to reduce noise and account for small inconsistencies in pose detection.

The extracted pose data must then be processed to obtain the angles for each joint on the target NAO robot, which we will solve by utilizing direct analytical geometry methods. The procedure followed in our approach is shown in Fig. 2.

1) 3D Human Pose Estimation

To perform 3D Human Pose Estimation, we utilize Google's Mediapipe Pose solution. Mediapipe is a library that contains several machine learning solutions for media processing. The solution used is called BlazePose (Fig. 3), which is a lightweight convolutional neural network for 3D human pose estimation that is targeted to mobile devices and developed by the Google Research

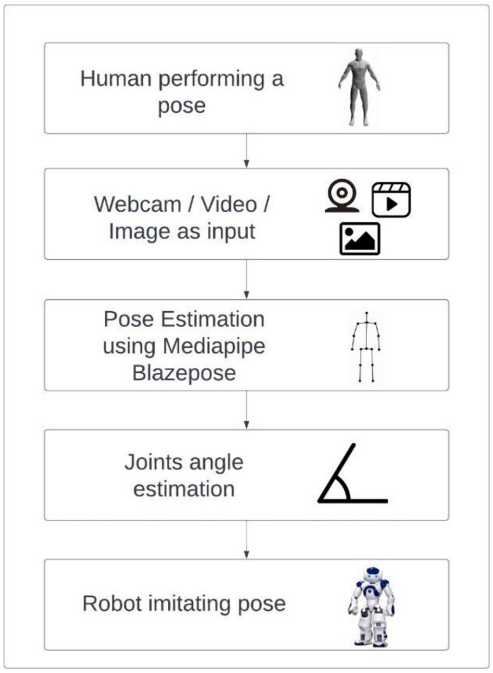


Fig. 2. Flowchart detailing every step in our approach.

Fig. 3. BlazePose results [15].

Team [15].

This model is ideal for our purposes, as it achieves outstanding performance using CPU alone, to the extent that it can handle high frame per second videos and even approach the performance of other state-of-the-art solutions using GPU [16]. The model approach involves using heatmaps and regression to match the keypoint

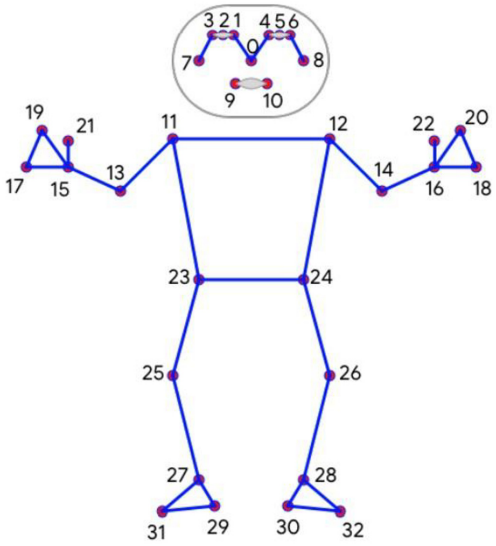


Fig. 4. BlazePose 33-keypoint topology [15].

coordinates. The topology of the estimation contains 33 keypoints as shown in Fig. 4.

2) Calculating Angles from the Detected Pose

To begin, we developed a simple demonstrative example of motion retargeting utilizing 2D human pose estimation alone, along with trivial vector operations to calculate only shoulder and elbow roll from 2D keypoints.

We obtained satisfactory results for two dimensions and decided to attempt to generalize it further into three dimensions, which proved to be conceptually challenging, but mostly feasible.

For the explanation of our method, we will employ the definitions from Table 1.

Generating the reference axes: The first step of our method is to generate some referential axes for the detected pose that we will later use to calculate some joint angles on the detected pose.

Initially, we calculate the pose's right axis $\vec{R}$ as the simple subtraction of the location of the left shoulder from right shoulder.

$$\vec{R} = S_r - S_l$$

Then, the pose's forward axis $\vec{F}$ will be calculated as the average of the normal vectors of the triangles that can be formed from the torso keypoints.

Every tuple of three adjacent torso keypoints forms a well-defined triangle T (i.e., $T = H_l, H_r, Sr$). Thus, the normal vector for each of the triangles can be calculated as the cross product of the vectors $(T_3 - T_2) \times (T_1 - T_2)$ (the triangles in Fig. 5). The forward vector $\vec{F}$ is assigned to be the average of the four torso normal vectors.

Finally, the up axis can be calculated as the cross-

Table 1. Symbol definitions (assume all direction vectors are taken with unit-size, even if not specified)

Symbol	Definition
S_r	Location of the detected right shoulder joint
H_r	Location of the detected right hip joint
E_r	Location of the detected right elbow joint
W_r	Location of the detected right wrist joint
S_l	Location of the detected left shoulder joint
H_l	Location of the detected left hip joint
E_l	Location of the detected left elbow joint
W_l	Location of the detected left wrist joint
$\vec{V}_{SE}$	Direction vector from an arm's shoulder to elbow
$\vec{V}_{EW}$	Direction vector from an arm's elbow to wrist
$\vec{V}_{WI}$	Direction vector from an arm's wrist to its index finger
$\vec{V}_{WP}$	Direction vector from an arm's wrist to its pinky finger
$\vec{F}$	Referential front-axis vector for the detected pose
$\vec{R}$	Referential right-axis vector for the detected pose
$\vec{U}$	Referential up-axis vector for the detected pose

Fig. 5. Torso triangles for forward-vector calculation.

product of the right vector $\vec{R}$ and forward vector $\vec{F}$ as follows:

$$\vec{U} = \vec{R} \times \vec{F}$$

Shoulder pitch and roll: It tends to be difficult to correctly estimate the shoulder angles from detected keypoints, because a composed rotation is needed to properly represent the human shoulder (pitch and then roll, or vice-versa).

We calculate the shoulder pitch as the angle between the vectors $\vec{V}_{SE}$ and $\vec{F}$ when projected onto the plane generated by the right-axis vector $\vec{R}$ (that is, the plane to which $\vec{R}$ is orthonormal) (Fig. 6).

Then, to calculate shoulder roll, first we have to calculate the plane over which the shoulder roll will perform its movement. The orthogonal vector that defines this plane can be simply calculated as the cross-product of the side-projection of $\vec{V}_{SE}$ with the reference lateral axis (right vector). $\vec{V}_{SE}$ is then projected onto said plane, after which the shoulder roll can finally be trivially calculated as the angle between $\vec{V}_{SE}$ and the reference lateral axis (both of which are projected onto the shoulder-roll plane) (Fig. 7).

Elbow roll: Elbow roll may be calculated as the simple angle between the elbow-shoulder, which is equal to $-\vec{V}_{ES}$, and the elbow-wrist vector $\vec{V}_{EW}$. Given vectors $\vec{V}_{ES}$ and $\vec{V}_{EW}$, the angle between them is calculated by taking their cross-product as the reference vector for the angle sign (Fig. 8).

Elbow yaw: Initially, we tried to calculate the elbow yaw by transforming the reference forward and right vectors $\vec{F}$ and $\vec{R}$ with the successive relative transform of the shoulder and elbow joint, after which we calculated the elbow yaw by projecting the elbow to wrist vector $\vec{V}_{EW}$ onto the plane of the transformed vectors.

However, we decided against implementing this method directly as it would likely be very error-prone due to its heavy dependence on the precision of the estimation

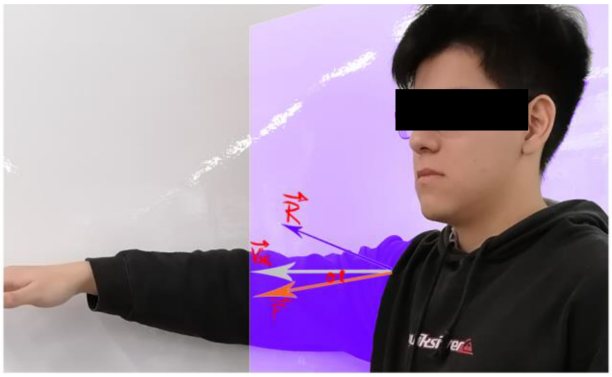


Fig. 6. Projection of $\vec{V}_{SE}$ and $\vec{F}$ onto $\vec{R}$'s plane.

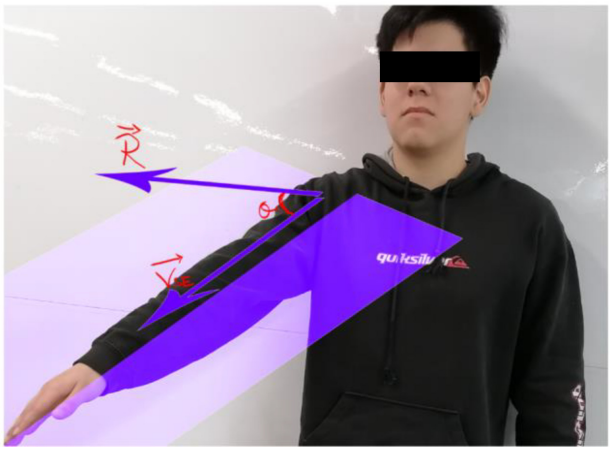


Fig. 7. Plane formed by $\vec{R}$ and $\vec{V}_{SE}$, and the angle formed by the two vectors.

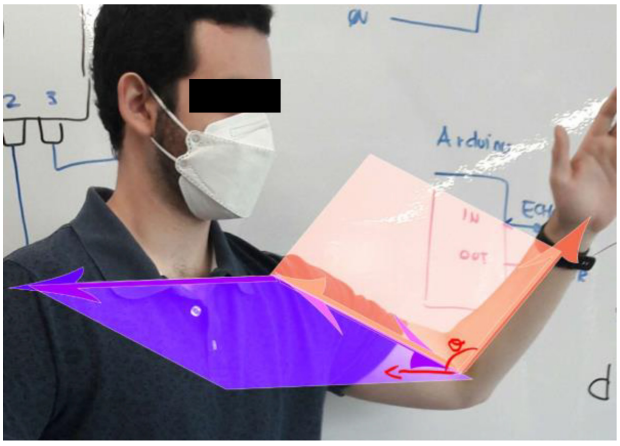


Fig. 9. Elbow yaw calculation. Visualization of the elbow plane and shoulder-elbow plane (purple) and the shoulder-elbow-wrist plane (orange).

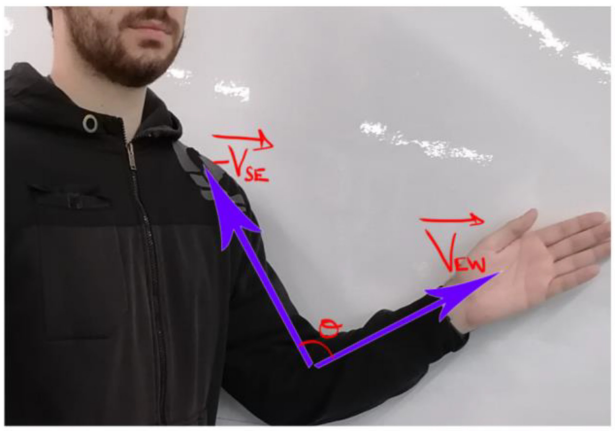


Fig. 8. Elbow roll calculation with angle between $\vec{V}_{EW}$ and $-\vec{V}_{SE}$.

of the locations and transformations of the shoulder and elbow joints, and it would also require more complex vector and matrix operations.

Therefore, we ended up reformulating our idea in a very similar manner, but while utilizing the plane generated by the shoulders and elbow joint (that is, the plane defined by vectors $\vec{R}$ and $\vec{V}_{SE}$, respectively) instead.

To actually calculate the angle of the vector $\vec{V}_{EW}$ relative to the aforementioned plane, it is easier to calculate the angle between the shoulders-elbow and shoulder-elbow-wrist planes (Fig. 9).

The angle between two planes is equivalent to the angle between their normals, so the final elbow yaw can be simply calculated as the angle between the vectors $(\vec{R}\times\vec{V}_{SE})$ and $(\vec{V}_{EW}\times-\vec{V}_{SE})$. Even though it was derived using a different approach, this resulted in a formulation equivalent to the one utilized by [17].

Wrist yaw: For the calculation of the wrist yaw, it is possible to employ a very similar method to that used for the elbow yaw. We utilize the shoulder-elbow-wrist plane as the base reference, and we calculate a vector normal to the palm of the hand by utilizing the hand's index and pinky finger joints. Specifically, the normal vector of the back-side of the palm can be calculated as the cross-product of $\vec{V}_{WP}$ and $\vec{V}_{WI}$. With this, we can trivially calculate the angle between the hand-plane and the shoulder-elbow-wrist plane as the angle between their normal vectors, in the exact same manner as was explained for the elbow yaw calculations.

Head pitch and yaw: To calculate the corresponding head angles, we will generate a frontal vector for the face of the detected person. This vector is generated as the mean of two vectors: the normal vector of the triangle formed by the nose and eyes as well as the normal vector of the triangle formed by the nose and both endpoints of the mouth.

With the face frontal vector, we can simply compare this to the torso frontal and calculate the angles projected onto the lateral plane (pitch) and the "floor" plane (yaw). Once again, the lateral plane is defined simply by the right vector that was initially calculated for the detected pose. The "floor" plane is also defined by the calculated up-vector.

Averaging over frames: The last step, after having calculated all the joint angles from the detected poses, is to calculate an average of the new set of angles with the angles obtained from previous frames. We found that, in practice, averaging the angles over the last three frames yields a good balance of smooth and accurate motion replication, so this is the value we utilize by default. However, in the experiments section, we attempt to use different numbers of frames for the average (from 1 to 4) and explore the corresponding effect on the accuracy of pose replication.

IV. EXPERIMENTS

In this section, we present and explain the experiments we have performed with our proposal, how to replicate them, and the results we have obtained.

For the experimentation, we recorded a couple of motions that involved movement of the upper body joints. Table 2 lists each motion used for our approach. The videos themselves can be found on the following Google Drive folder: https://drive.google.com/drive/folders/12O1Vt3utwt01d1WI3Y-MRD8rKdvulQFW?usp=share_link.

1) Experimental Protocol

The experiment was performed on a 16 GB RAM Ryzen 3 3100 4-Core 3.59 GHz PC. The data utilized for the experimentation was manually constructed and consisted of five monocular videos in which a person performs an upper body motion. All the utilized code can be found in the following GitHub repository: https://github.com/oscarburga/mrmime2.

2) Performance Evaluation

To evaluate the performance of our solution, we have to consider four different main elements that will contribute to the execution time: reading the data, running the pose estimation model, calculating the angles for the NAO robot, and sending the angles to the NAO robot.

For our purposes, we skip measuring the time it takes to read the input data (i.e., the image, frame of a video, or frame of a webcam stream), as well as the time it takes to send the pose information to the robot. Therefore, we only measure the time taken to perform pose detection with Mediapipe's BlazePose, as well as the time taken to generate the angles from the detected pose data. This information is calculated per-processed-frame, and we present the average obtained for each of the videos in Table 3.

Table 2. Motions used for experimentation

	Action performed
Motion 1	Waving of arms
Motion 2	Raising of arms
Motion 3	Arms stretching forward
Motion 4	Arms stretching to the sides
Motion 5	Rotation of elbows

Table 3. Average time taken to process each frame (unit: second)

	Average time (s)		
	Pose	Angles	Total
Motion 1	0.02949	0.00188	0.03137
Motion 2	0.02938	0.00184	0.03122
Motion 3	0.02965	0.00185	0.03122
Motion 4	0.02944	0.00201	0.03145
Motion 5	0.02942	0.00192	0.03134

3) Experimental Comparative Evaluation

The target of our experimentation is to evaluate the performance of our approach in terms of accuracy and optimize it to achieve better results. To be able to measure the accuracy of our approach, we utilize a metric called Fréchet distance [18]. This metric was initially proposed to measure the similarities between polygonal curves while considering the order of points, and it has been utilized by several authors to measure the effectiveness of their motion retargeting solution [1, 19]. A smaller Fréchet distance indicates a stronger similarity between curves.

In our approach, the curves that are compared are the arms after the motion retargeting is applied. These are defined by three joints: the shoulders, elbows, and wrists. The head is not included because there is no translation involved in its motion. To be able to estimate the overall effectiveness in three dimensions, we evaluate each plane independently is an *n*-dimension approach that can be used to obtain the Fréchet distance, but other authors tend to make comparisons using only two dimensions [1, 19].

We experienced some difficulties when trying to design fair comparisons for our solution with those presented by other authors, because there are not many open approaches that utilize human pose estimation on monocular videos or images. Most of the solutions utilize more than one camera or sensors to obtain human joint positions, which benefits their results and also makes and then average each of the results. The data compatibility made it harder for us.

We were not able to find any existing open solution that utilized the NAO robot and worked only with monocular images, but there were a couple of solutions that used Fréchet distance as an evaluation metric. Even though the environment and complexity of those approaches differ from ours, they will help us define the Fréchet distance boundaries of a good solution. The authors in [1] developed a motion retargeting solution for a YUMI robot and obtained an overall Fréchet distance of 0.13 after evaluating several motions. The authors collected popular works and compared their Fréchet distances with the one they obtained. Overall, the Fréchet distances of the mentioned works were between 0.14 and 0.20, and this range can be used as an indicator of good results [1].

We executed our approach using five monocular videos in which a person performs an upper body motion focused on the arms. Our variable on each execution of the five motions was the number of frames used to

Table 4. Comparative table of results by human pose estimation framerate

	Number of frames to average							
	1 frame		2 frames		3 frames		4 frames	
	Lt. arm	Rt. arm	Lt. arm	Rt. arm	Lt. arm	Rt. arm	Lt. arm	Rt. arm
Motion 1	0.43	0.44	0.43	0.44	0.43	0.44	0.43	0.44
Motion 2	0.43	0.47	0.43	0.47	0.43	0.47	0.43	0.47
Motion 3	0.43	0.44	0.43	0.44	0.43	0.48	0.43	0.48
Motion 4	0.43	0.43	0.47	0.43	0.43	0.47	0.43	0.47
Motion 5	0.42	0.42	0.47	0.42	0.42	0.47	0.42	0.47
Overall mean	0.43	0.44	0.45	0.44	0.43	0.47	0.43	0.47

calculate the averages for the angles generated from the last detected pose. We measured the Fréchet distance between the curves of the arms independently for each arm and every frame and calculated the mean of all these results. The obtained results are presented in Table 4.

The results our approach obtained after the experimentation process were not optimal compared to those obtained by other works. As can be seen in Table 4, the Fréchet mean for each video is greater than 0.43, which is inferior to those of other state-of-the-art solutions.

There are several factors that can help explain these results. One of them is related to the human joints calculation, which involves using human pose estimation based on monocular video alone, and this involves a lack of depth information when obtaining the human pose, specifically that the elbow placement was typically in front of the person's torso even if their arms were straight. All the monocular videos were taken from a perspective in front of the person, which made the depth calculations harder.

We also noticed that the right shoulder presented a slight forward inclination that caused additional differences after the motion retargeting was performed. This is evidenced in the higher Fréchet distance for the right arm, as can be seen in Table 4.

It is worth mentioning that, while we obtain the lowest Fréchet distance while using only one frame on average, in practice we can empirically observe that the movements tend to be smoother and less violent when averaging between 2–4 frames. The differences in Fréchet distance are also small when compared among these numbers of frames for averaging, so we can consider using 2–4 frames for averaging if we are interested in achieving a smoother, more natural motion over maximizing accuracy.

V. CONCLUSION AND PERSPECTIVES

In the present work, we proposed using 3D human pose estimation from monocular images and videos in combination with direct analytical geometry methods to perform motion retargeting from a human person demonstrator to the NAO robot. We implemented the proposed solution while utilizing the BlazePose 3D human pose estimation model from Google's Mediapipe library, and we also wrote our own geometric primitives and calculations mostly from scratch. Finally, we empirically evaluate our solution and performed both an analysis of the results utilizing the Fréchet distance metric as well as meaningful qualitative observations identifying the weaknesses and strong points of our solution.

We conclude that performing human-robot motion retargeting from monocular images and videos can be successfully achieved through utilizing computer vision methods for performing 3D pose estimation, as well as through the use of direct analytical geometry calculations to obtain the angles of the robot.

At present, having to rely on the correctness of the poses detected by the 3D pose estimation model is a significant weakness of approaches like ours. Depth ambiguities and body occlusions are very detrimental to the quality of the results. However, this only means that, as more research and development is conducted in the field of 3D pose estimation, approaches like ours will become passively better and more reliable over time.

We strongly believe that motion retargeting from monocular images and videos is, and will continue to be a huge step toward making robotics more accessible for both users and programmers alike. As such, for future works, we encourage fellow researchers to continue building motion retargeting solutions based on 3D human pose estimation from monocular imagery, not only to achieve better retargeting accuracy metrics but also to start branching out into more meaningful tasks and applications: leg motion replication (locomotion, walking, etc.), object manipulation (grabbing and letting go), etc. It would also be ideal to explore more beneficial applications of motion retargeting in different fields: healthcare, sports, tasks automation, and so on.

CONFLICT OF INTEREST

The authors have declared that no competing interests exist.

REFERENCES

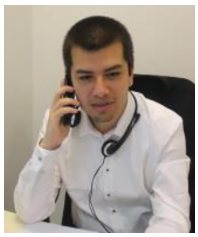
1. H. Zhang, W. Li, Y. Liang, Z. Chen, Y. Cui, Y. Wang, and R. Xiong, "Human-Robot motion retargeting via neural latent optimization," 2021 [Online]. Available: https://arxiv.org/abs/2103.08882v1.
2. R. Elbasiony and W. Gomaa, "Humanoids skill learning based on real-time human motion imitation using Kinect," *Intelligent Service Robotics*, vol. 11, pp. 149-169, 2018. https://doi.org/10.1007/s11370-018-0247-z
3. T. Kim and J. H. Lee, "C-3PO: cyclic-three-phase optimization for human-robot motion retargeting based on reinforcement learning," in *Proceedings of 2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 8425-8432. https://doi.org/10.1109/ICRA40945.2020.9196948
4. H. Khalil, E. Coronado, and G. Venture, "Human motion retargeting to Pepper humanoid robot from uncalibrated videos using human pose estimation," in *Proceedings of 2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, Vancouver, Canada, 2021, pp. 1145-1152. https://doi.org/10.1109/RO-MAN50785.2021.9515495
5. O. Ossmy and K. E. Adolph, "Real-time assembly of coordination patterns in human infants," *Current Biology*, vol. 30, no. 23, pp. 4553-4562, 2020. https://doi.org/10.1016/j.cub.2020.08.073
6. J. Wang, K. Qiu, H. Peng, J. Fu, and J. Zhu, "AI coach: deep human pose estimation and analysis for personalized athletic training assistance," in *Proceedings of the 27th ACM International Conference on Multimedia*, Nice, France, 2019, pp. 374-382. https://doi.org/10.1145/3343031.3350910
7. N. Blanchard, K. Skinner, A. Kemp, W. Scheirer, and P. Flynn, "Keep Me In, Coach!: a computer vision perspective on assessing ACL injury risk in female athletes," in *Proceedings of 2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, USA, 2019, pp. 1366-1374. https://doi.org/10.1109/WACV.2019.00150
8. W. Hu, C. Zhang, F. Zhan, L. Zhang, and T. T. Wong, "Conditional directed graph convolution for 3D human pose estimation," in *Proceedings of the 29th ACM International Conference on Multimedia*, Virtual Event, China, 2021, pp. 602-611. https://doi.org/10.1145/3474085.3475219
9. S. Wang, X. Zuo, R. Wang, F. Cheng, and R. Yang, "A generative human-robot motion retargeting approach using a single depth sensor," in *Proceedings of 2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017, pp. 5369-5376. https://doi.org/10.1109/ICRA.2017.7989632
10. K. S. Hwang, J. C. Jiang, Y. J. Chen, and H. Shi, "Motion segmentation and balancing for a biped robot's imitation learning," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1099-1108, 2017. https://doi.org/10.1109/TII.2017.2647993
11. J. M. Sa and K. S. Choi, "Humanoid robot teleoperation system using a fast vision-based pose estimation and refinement method," *IEIE Transactions on Smart Processing & Computing*, vol. 10, no. 1, pp. 24-30, 2021. https://doi.org/10.5573/IEIESPC.2021.10.1.024
12. X. Wu, C. Yang, Y. Zhu, W. Wu, and Q. Wei, "integrated vision-based system for efficient robot arm teleoperation," *Industrial Robot*, vol. 48, no. 2, pp. 199-210, 2021. https://doi.org/10.1108/IR-06-2020-0129
13. G. Lan, Y. Wu, F. Hu, and Q. Hao, "Vision-based human pose estimation via deep learning: a survey," *IEEE Transactions on Human-Machine Systems*, vol. 53, no. 1, pp. 253-268, 2023. https://doi.org/10.1109/THMS.2022.3219242
14. K. Ayusawa and E. Yoshida, "Motion retargeting for humanoid robots based on simultaneous morphing parameter identification and motion optimization," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1343-1357, 2017. https://doi.org/10.1109/TRO.2017.2752711
15. V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: on-device real-time body pose tracking," 2020 [Online]. Available: https://arxiv.org/abs/2006.10204.
16. Z. Cao, G. Hidalgo, T. Simon, S. E. Wei, and Y. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172-186, 2021. https://doi.org/10.1109/TPAMI.2019.2929257
17. A. Balmik, M. Jha, and A. Nandy, "NAO robot teleoperation with human motion recognition," *Arabian Journal for Science and Engineering*, vol. 47, no. 2, pp. 1137-1146, 2022. https://doi.org/10.1007/s13369-021-06051-2
18. H. K. Ahn, C. Knauer, M. Scherfenberg, L. Schlipf, and A. Vigneron, "Computing the discrete Fréchet distance with imprecise input," *International Journal of Computational Geometry & Applications*, vol. 22, no. 1, pp. 27-44, 2012. https://doi.org/10.1142/S0218195912600023
19. Y. Liang, W. Li, Y. Wang, R. Xiong, Y. Mao, and J. Zhang, "Dynamic movement primitive based motion retargeting for dual-arm sign language motions," in *Proceedings of 2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, 2021, p. 8195-8201. https://doi.org/10.1109/ICRA48506.2021.9561120

Oscar Burga https://orcid.org/0000-0003-3501-3533

Oscar Burga received his bachelor's degree in computer science from the Universidad Peruana de Ciencias Aplicadas, Peru in 2022. He works a developer for videgames companies that use artificial intelligence for their products. He is specialized in competitive programming and complex algorithms.

Jonathan Villegas

Jonathan Villegas received his bachelor's degree in computer science from the Universidad Peruana de Ciencias Aplicadas, Peru in 2022. He works as a data engineer at BBVA Bank. His research interests are in data visualization, data pre-processing and data mining.



Willy Ugarte https://orcid.org/0000-0002-7510-618X

Willy Ugarte received his Ph.D. degree in computer science from the University of Caen Normandy, France in 2014. He did a post-doctoral study at the University of Grenoble Alpes, France, and then joined the Universidad Peruana de Ciencias Aplicadas, Peru in March 2018. Dr. Willy Ugarte also supervises Ph.D. candidates from Universidad Nacional Mayor de San Marcos, Peru. His research interests are in artificial intelligence, machine learning, constraint programming, solvers, data mining and cross-fertilization among these fields.