Using a 1D Pose-Descriptor on the Finger-Level to Reduce the Dimensions in Hand Posture Estimation

Amin Dadgar and Guido Brunnett
Computer Science, Chemnitz University of Technology, Straße der Nationen 62, 09111, Chemnitz, Germany

Keywords: One-D Finger Pose-Descriptor, Distance-Based Descriptor, Anatomy-Based Dimensionality Reduction, Temporal a-Priori, Hand Posture Estimation, Single RGB Camera, Virtual Hand Models, Computer Vision.

Abstract: We claim there is a simple measure to characterize all postures of every finger in human hands, each with a single and unique value. To that, we illustrate the sum of distances of fingers' (movable) joints/nodes (or of the finger's tip) to a locally fixed reference point on that hand (e.g., wrist joint) equals a unique value for each finger's posture. We support our hypothesis by presenting numerical justification based on the kinematic skeleton of a human hand for four fingers and by providing evidence on two virtual hand models (which closely resemble the structure of human hands) for thumbs. The employment of this descriptor reduces the dimensionality of the finger's space from 16 to 5 (e.g., one degree of freedom for each finger). To demonstrate the advantages of employing this measure for finger pose estimation, we utilize it as a temporal *a-priori* in the analysis-by-synthesis framework to constrain the posture space in searching and estimating the optimum pose of fingers more efficiently. In a set of experiments, we show the benefits of employing this descriptor in time complexity, latency, and accuracy of the pose estimation of our virtual hand.

1 INTRODUCTION

Three-D hand pose estimation systems aim at detecting the joint configuration of human hands in 3D space. These systems are essential requirements for disciplines such as human behavior understanding, human-computer interaction, and augmented reality. However, the high degrees of freedom of fingers (16 out of total 28 DoF of hands) is cumbersome for a fast and/or accurate performance. Therefore, it is advantageous to discover the feasibility of reducing this high dimensionality by exploiting the hands' inherent kinematic/anatomic properties.

The main idea of our work is to investigate whether a finger pose could be uniquely described as a distance between the keypoints on fingers and a locally fixed reference point on the hands (e.g., the wrist joint, palm center, etc.). Such a relation would simplify the representation of the fingers' postures to merely five numbers and thus drastically reduce the dimensionality of the problem.

Similar approaches have been suggested in two-D (Liao et al., 2012) though the authors failed to generalize them to three-D. There were endeavors on three-D (Shimizume et al., 2019) but their generalizations to all postures and orientations remained challenging. Thus in those works, simplifying the problem, by modeling the camera-hand distance, assuming stretched outward fingers, and strictly fixating the hand orientation, seems to the expected trend.

In our investigation, by employing two artificial hand models, we were able to observe two different yet highly related *pose-descriptors* (Eq1, 2). First, the distance of the fingers or thumb's tip to a reference point (wrist) is always a unique value for different poses, Fig1-middle. Second, the sum of distances of fingers/thumb (movable) joints/nodes to that reference point is also a unique value, Fig1-right.

After validating the uniqueness of the *pose-descriptors* for both models, we extended our investigation to justify our findings based on the kinematic relations of the human hand. In the Section 3, we numerically justify, given a range of the allowable rotation for joints, if one merely selects the place of the reference point carefully, the uniqueness of these distances will be maintained. We label the suitable spot of reference points in Fig3 as Ref_f.

In brief, we require the following components to compute and assess the uniqueness of our *pose-descriptors*: A) Positions of moving nodes of fingers (e.g., two inter-phalangeal joins and the tip), and the thumb's (i.e., two phalangeal joints and the tip) as

Dadgar, A. and Brunnett, G.
Using a 1D Pose-Descriptor on the Finger-Level to Reduce the Dimensions in Hand Posture Estimation.
DOI: 10.5220/0011781000003411
In *Proceedings of the 12th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2023)*, pages 501-508
ISBN: 978-989-758-626-2; ISSN: 2184-4313

seen in Fig1-top. B) Appropriate reference point that always remains locally static with respect to fingers (e.g., the wrist joint). C) Plausible rotation range of each movable node. This *descriptor* is independent of camera distance to the hands, hand orientation (or camera view-point), and fingers pose.

We employ this *pose-descriptor* to reduce the high degrees of freedom of fingers to five (one-D for each finger). Additionally, we will incorporate this as a-priori in five one-D temporal models (one model for each finger) and achieve a real-time estimation of the finger poses of a virtual hand in the costly synthesis-by-analysis framework on CPUs.

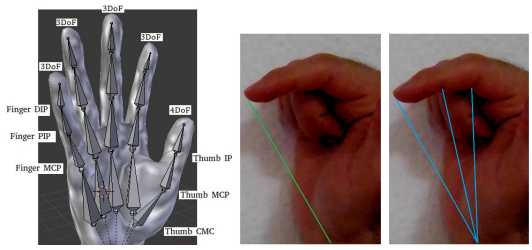

Figure 1: Left: CMC carpal-metacarpal joint, MCP metacarpal- phalangeal joint, PIP proximal inter-phalangeal joint, DIP distal interphalangeal joint, IP inter-phalangeal joint. Both of these values, Middle: the distance of fingers' tip to a local reference point (e.g., wrist joint) and Right: The sum of the distances of 3 movable fingers' nodes, on a hand gives us a unique value for each finger posture.

2 LITERATURE REVIEW

For accurate and real-time estimation of the hand's three-D postures, researchers consider a wide range of approaches (Zhang et al., 2020; Zhao et al., 2013). To review the fore works regarding global relationships between the fingers and a point on the hands, however, we focus on a specific type of posture estimation that detects fingertips position. As one of the earliest works in finding some global features on fingers, (Hardenberg, 2001) demonstrated a circular-diametric relationship between the fingers on two-D. Then he tried to find the fingertip's position based on those relationships.

There are several distance-based approaches to detecting fingertips in two-D. The work by (Dung and Mizukawa, 2010) suggested a distance-based method in connected component labeling (Paralic, 2012) fashion to extract the two-D fingertips, hand region, and palm center on images. However, their approach works only if the fingers are wide open. A more advanced strategy proposed by (Liao et al., 2012) addresses more challenging hand postures such as closed-fingers poses. They employed distance transformation to filter fingers and remain with merely the palm area. However, they used a simplified two-D hand model with strongly local geometric constraints.

These approaches have two main drawbacks that are intrinsic to their two-D characteristics. First, they rely on local properties that are assumed will remain unchanged. However, these two-D properties inherently are against the innate three-D global invariability (unless the hand faces forward, with a specified distance to the camera and a fixed orientation). Second, their extensions to other scenarios and applications are not straightforward.

To alleviate the problems above, several three-D distance-based approaches were proposed. For instance, amid the depth sensors' era, (Raheja et al., 2011) proposed a method for fingertips and centers of palms detection using KINECT. However, they assumed that fingertips are closer to the camera than the other hand's parts, which imposes brute constraints on the orientation of hands.

By incorporating monocular sensors and proposing a novel finger constraint, (Yamamoto et al., 2012; Shimizume et al., 2019) estimated hand postures using detected fingertip positions with inverse kinematics (*IK*). However, there is a major issue in their approach. In *IK*, there are relations from fingertips to the joints' configuration (not the reverse). Therefore, in practice, one can not meaningfully reduce the high complexity (or DoF) of fingers. For example, an efficient temporal model can not be introduced on the *IK*. Thus the system has to estimate hand postures from fingertip positions by solving IK for a high DoF problem. That novel constraint was employed to identify the touch state of the fingers' tips. However, the extension of the approach for every posture of the hand seems infeasible.

To the best of our knowledge, we propose a novel distance-based measure to describe the poses of each finger. Our descriptor is in three-D and is view-point and camera-distance independent. Therefore, it can assist one in designing finger pose estimation systems with merely five DoF.

3 METHODOLOGY

Our *pose-descriptor* for each finer is a one-dimensional number that characterizes the postures of that finger. To compute this descriptor we require the distance of fingertips (and movable finger joints) to a locally fixed point on the hands (e.g., the wrist joint). In this section, we demonstrate how we calculate this *descriptor* and justify its uniqueness within plausible ranges of fingers kinematics (degrees). Additionally,

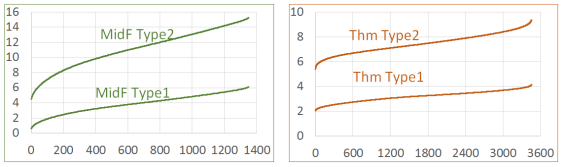

Figure 2: The sorted *pose-descriptor* $Type_1$ (Eq1) and $Type_2$ (Eq2), for middle finger & thumb. The four fingers have the same number of DoF and a similar descriptor pattern. We illustrate the Mid-Res database consisting of ≈1400 poses for the finger and 3600 poses for the thumb.

we show how to incorporate it in a temporal model as *a-priori* and design a search engine to estimate the poses of a virtual hand in real-time.

Calculation of a Novel Pose-Descriptor. The basis of our calculations is a data structure that comprises the positions of all finger joints and tips. Considering 21 joints in hands, we define 7 limbs Lt, Rn, Md, Id, Th, Wr, and Fr which stand for little, ring, middle, index, thumb, wrist, and forearm, respectively. Additionally, each finger has a tip t node and u,m,l,p joints which stand for upper, middle, lower, and palm (note the thumb has no middle joint).

By choosing the wrist as the Ref point, the fingertips distances to the wrist are calculated as follows, and call it the *pose-descriptor* $Type_1$:

$$\begin{aligned} Descriptor_1(Fin_i) &= D(Tip(Fin_i), Wr) \\ where,\ \ Fin_i &= \{Lt, Rn, Md, Id, Th\} \end{aligned} \tag{1}$$

We also define the $Type_2$ descriptor as the sum of the movable nodes' distances to that Ref point.

$$\begin{aligned} Descriptor_2(Fin_i) &= \sum_{Node=t,u,m} D(Node(Fin_i), Wr) \\ where,\ \ Fin_i &= \{Lt, Rn, Md, Id\} \\ Descriptor_2(Th) &= \sum_{Node=t,u,l} D(Node(Th), Wr) \end{aligned} \tag{2}$$

In this equation, all fingers have the joint's indexes of t,u,m, and the thumb joint's indexes are t,u,l.

In our experiments, we investigate the advantages of the second type of this *pose-descriptor*. However, as the following paragraphs will clarify, both descriptors are unique. Thus in some applications, where the position of other joints is not known, the first type would work just as fine.

Pose-Descriptor Uniqueness: Justification & Evidence. To employ this metric as a *pose-descriptor*, we *justify* it has a unique value for each finger's most (or ideally all) poses for an excessively large number of finger posture set (e.g., high-resolution pose database). To simplify the investigation, as seen in Fig3, we begin with the hand kinematics for *pose-descriptor* $Type_2$. Also, we initially consider the distance of the fingers moving nodes to the palm joint of each finger, *PalmJ* (not the wrist joint), to eliminate the calculations on the Y-axis (using Eq3). Next, we set the ranges for the finger joints' angles. For the four angles of the fingers, θ_1, θ_2, θ_3, and θ_4 we observe the ranges of $[0°,0°]$ (e.g., L_1 is a fixed part), $[0°,90°]$, $[0°,120°]$, and $[0°,45°]$, respectively.

$$\begin{aligned} Z &= Z_T + Z_U + Z_M = (z_1+z_2+z_3+z_4) + \\ &\quad (z_1+z_2+z_3) + (z_1+z_2) = 3z_1 + 3z_2 + 2z_3 + z_4 \\ &= 3L_1 cos(\theta_1) + 3L_2 cos(\theta_2) + 2L_3 cos(\theta_3) + L_4 cos(\theta_4) \\ X &= X_T + X_U + X_M = (x_1+x_2+x_3+x_4) + \\ &\quad (x_1+x_2+x_3) + (x_1+x_2) = 3x_1 + 3x_2 + 2x_3 + x_4 \\ &= 3L_1 sin(\theta_1) + 3L_2 sin(\theta_2) + 2L_3 sin(\theta_3) + L_4 sin(\theta_4) \end{aligned} \tag{3}$$

Then considering $Theta_i$s, we have a function on the hyperspace in which we can form a parametric line (with parameter t) that lies on the intersection of this function and an arbitrary plane. Now, for every value of t and $Theta_i$, if the derivative of this line is non-zero, we can conclude Eq3 is injective (See appendix for developing this derivative). To perform a thorough numerical justification, we also investigate the influence of the bones' length and calculate the derivative for different length values. To that, we normalize L_is w.r.t L_2 (the lowest 'moving' limb on each finger) and alter the values of others ($L_i > 0$). Finally, we calculate this derivative for any two points on the high-resolution database of *1-degree-step* for each joint (e.g. $486K$ poses).

In our investigations, the smallest value the derivative gets mostly is in the order of 10^{-9} and rarely to 10^{-16}. Considering the precision of python as 10^{-28}, we can conclude these smallest values are non-zero (even with $L_2 = L_3 = L_4 = 1$ only if $L_1 \neq L_2$). Thus, Eq3 is unique for all poses of fingers in our excessively high-resolution database, and in the uniqueness property of the descriptor, the fixed part's length has the most significant role. One can select *PalmJ* in a spot where the condition of $L_1 \neq L_2$ satisfies.

The next step is to incorporate the Y-axis distance into the computation, and to show the *pose-descriptor* stays unique if the reference point is *not* on the z-axis of the fingers. As demonstrated in Fig3-right, $Y_T^2 = Z_T^2 + ConstY^2$, $Y_U^2 = Z_U^2 + ConstY^2$, and $Y_M^2 = Z_M^2 + ConstY^2$. Therefore, the total distance on Y-axis is $Y^2 = Z_T^2 + Z_U^2 + Z_M^2 + 3 \times ConstY^2$. However, one term of this equation is constant, and the Z terms are the previously calculated Z-distance. Therefore, we conclude that our justification is extendable to an arbitrary $RefJ$ on the Y-axis. The same conclusion is derivable for $Type_1$ *descriptor* by conducting a similar analysis. Because by a closer look at the Eq3,

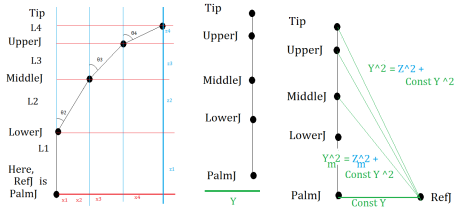

Figure 3: The kinematics formulation to calculate *pose-descriptor* $Type_2$: The left image shows the side view of a finger, to find Z and X values. The middle image demonstrates the case when $Y = 0$ and $D_{PalmJoint} = \sqrt[2]{X^2+Z^2}$. The right image illustrates that the incorporation of the Y-distance ($D_{PalmJoint} = \sqrt[2]{X^2+Y^2+Z^2}$) into the formula for *pose-descriptor* $Type_2$ and how no new variable enters the equation.

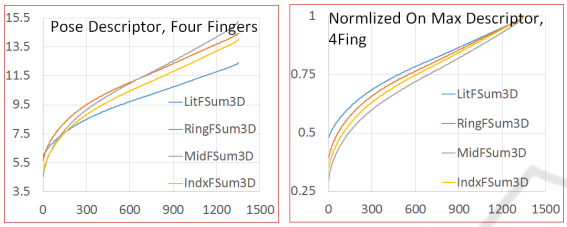

Figure 4: The actual and normalized *pose-descriptor* $Type_2$ values of four fingers. Thumb has a different number of poses thus its visualization with other fingers is not possible.

$Type_1$ is a special case of $Type_2$ where the z_i and x_i ($i \in \{1,2,3,4\}$) coefficients equal one.

Finally, we extend the uniqueness assessment to the thumbs. Thumbs, similar to fingers, have three moving nodes. However, unlike other fingers, there is no inherent fixed part in the thumbs' bone kinematics. Nevertheless, by defining the $RefJ$ somewhere below its $PalmJ$, we can assume a fixed limb for it. Therefore, a similar hierarchical structure of fingers is imitable for thumbs. However, a more significant discrepancy here is, fingers have three degrees of freedom while thumbs have four with entirely distinct ranges of degrees ($[-20,90]$, $[-10,10]$, $[-30,10]$, $[-20,20]$) and complex motion's structure. Arguing a similar kinematic formulations for thumbs (in the appendix) exceeds the limits for this paper.

Therefore to justify the uniqueness of thumbs, we employ two synthetic hand models with appropriate rigging and skinning. After using the defined pose representation and setting the virtual camera parameters, we compute the joints' three-D position. Then, we select a Low_{Res}, Mid_{Res}, and $High_{Res}$ database for each joint (e.g., less than 10°-step, more than $< 10^\circ$-step, 4°-step, respectively). These resolutions lead to a database of 300+, 3400+, and $>$ 32500 postures for the thumb on one of the hand model (for the second model, these numbers are slightly less as the degree ranges for that model is different). Then, we calculate the *pose-descriptor* $Type_1$ and $Type_2$, as explained

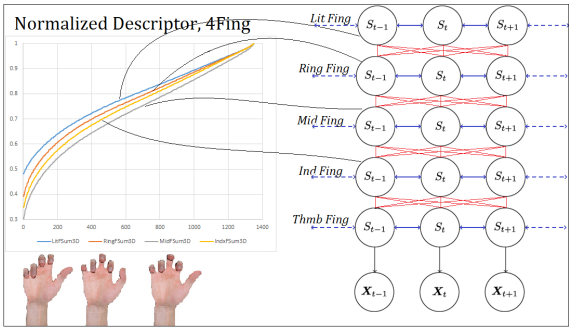

Figure 5: Various poses of all five fingers can relate together using *five* one-dimensional temporal model. Thumb has a different number of poses thus with could not visualize it's sorted *pose-descriptor* with other fingers.

in the previous subsection, and sort the values. Similar to the fingers, we witness unique *descriptor* values for all different resolutions of thumb postures for both types of descriptors and on both models (See Fig2 for $Type_1$, and for $Type_2$ on the first model).

In real-life applications (and our experiments), the number of considered poses is usually much smaller than the considered ones. Therefore, those large posture sets provide a high level of confidence about the uniqueness of our proposed descriptor.

One-D Temporal Model. Initially, we sought to illustrate the advantages of the *pose-descriptor* in the mere estimation of the finger poses (without considering the time complexity). Conducting a set of experiments, we achieved accurate pose estimations for *one* finger where the gradient descent utterly failed. However, the time complexity of the approach was high, thus we altered the roadmap and considered real-time performance as a crucial criterion in our evaluations.

To that, we utilize our descriptor in a one-D temporal Markovian model. As seen in Fig4, the model uses the previous and current states to estimate the next state. The figure visualizes all four fingers' one-dimensional *pose-descriptor*. Thumb has a different number of poses thus, simultaneous visualization of its descriptor in that figure is not possible.

Hands, as high DoF objects, require high dimensional (and complex) temporal model. However, with our *pose-descriptor* we can employ *five* 1D models to enhance the time-complexity considerably (Fig5). If n is the number of states and h is the number of searched fingers, the search algorithm will have $O(n^h)$ complexity. In a mid-resolution pose database, without the temporal model and considering merely the generate-and-test search strategy (GAT), n will be 1500(for four fingers) or 3500 (for thumbs). However, if we employ the temporal model, n will be 3, which leads to considerable time improvements.

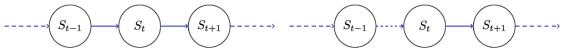

Figure 6: $GAT_{Dir_{2st}}$ (left) and $GAT_{Dir_{1st}}$ (right) to constraint the temporal model of five fingers based on the previous movement direction (open/closing) of fingers motion.

Further Improvement of Search Time. For finding the optimal pose, our approach is to compare the contours of the projected 3D model (into the 2D plane, S) to the contour of the input image (I) using Chamfer Distance (CD). However, CD is a costly procedure, because for each point on I, CD computes its distance to *all* points in S to find the minimum (Eq4)

$$CD = \frac{1}{|I|} \sum_{i \in (0,I)} Min_{s \in (0,S)}[d(I_i, S_s)] \quad (4)$$

However, our S and I are the 'sorted' (ordered) contour points' coordinates, so we can modify and speed up the Chamfer computation (Eq5): By solely calculating the distance around the neighborhood (nn) of the previously found point. That reduces the time complexity from $O(n^2)$ to $O(h*n)$. For the first point on I, we consider the entire points of S (initialization). In our experiments, nn is usually 5.

$$CD = \frac{1}{|I|} \sum_{i \in (0,I)} Min_{s \in (s-nn, s+nn)}[d(I_i, S_s)] \quad (5)$$

Real-Time Performance. Though these improvements considerably enhanced the time complexity, the performance is not real-time. Because in the temporal model, the GAT algorithm has $O(n^h)$ complexity where, thanks to our one-D temporal model, $n = 3$. That is, at each instance of time (S_t), there are forward, backward, and self transitions states each temporal model could undergo (Fig5). By considering five fingers, the overall complexity will amount to $n^h = 3^5 = 243$ poses to find the solution for one frame.

If we constrain them only to the left-to-right direction (as in Fig6-left, for example), at S_t, the transition can not go backward. So coming from S_{t-1}, it can either remain on S_t or go forward. That is, there are only two possible states each finger can undertake ($GAT_{Dir_{2st}}$), equating the total number of complexity to $n^h = 2^5 = 48$. A similar strategy exists for Fig6-right. If merely forward transition is possible ($GAT_{Dir_{1st}}$), the number of poses the algorithm would search equals to one: $n^h = 1^5 = 1$.

Three Motion Patterns. With these considerations, real-time estimation even on CPUs is feasible. Now, we need application-specific scenarios to allow the search, at each time step, to eliminate one or two states (out of three possibilities). To that, we introduce three motion patterns we observed in the fingers/thumb which assist us designing our scenarios.

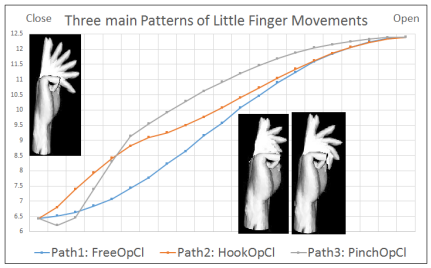

Figure 7: Three motion patterns of the little finger. We use them to design application-dependent scenarios. These patterns can also be very useful to analyze and model the time series of fingers' motion.

Four fingers have similar movement patterns but have different ones compared to the thumbs. As shown in Fig7 (using our Descriptor $Type_1$ for the little finger), there are three possible paths the four fingers can undertake to move between open and close states. First, is *freestyle* motion, denoted as $path_1$, in which all the finger's joints move together. Second, *hook-like* motion ($path_2$) in which, during the closing, the lowest joint of the fingers moves at the final phase, but first, other joints close until their limits. Third, *pinch-like* motion ($path_3$), in which during the closing, the lowest joint closes at first until its limit, then the other joints close together.

For the thumb with upper, lower, and palm joints, the $path_1$ is the *freestyle*, in which all the joints close together. For the $path_2$, (*stretching* motion), the thumb initially approaches the index finger by moving the lower joint. Then by moving the palm joint, the tip accosts the other fingers' parent joint. Next, by moving the upper joint, it closes the upper part, and finally, it ends the motion by returning the palm joint to its resting position. The $path_3$, (*wiping* motion), starts by moving its lower joint to approach the index finger (similar to the second path). Then, it closes the upper joint while being close to the index finger. Finally, it brings the lower joint to its resting position.

4 EXPERIMENT

We test the efficacy of our descriptor in finding the optimal solution in three experiments on a virtual hand as input. We use the $path_1$ to design application-specific scenarios with considerations from Table 1.

We employ four metrics to evaluate the performance. First is the time complexity, indicated by fps. Second is the initial and total latency caused by the initialization or unexpected costs in the middle of the estimation. Third is the average 3D Joint Position er-

Table 1: Eleven Variables could be considered in designing the scenarios that contain fingers' motion.

Exp Variables	
Do the fingers move A/Synchronously?	
Do they move orderly: Adjacent/Apart?	Is the order Right-to-Left?
If failed, do we Srch Adj Fings Only?	Is fingers' Motion Full-Cycle?
If not, do they Conserve their Direction?	Is there Hand-lvl Open-Closing?
Do we consider Previous Direction?	If yes, is it Initialized?
No. of Prtl-Cycled Finger's Dynamics? 1, 2, or 3?	
No. of End-Cycled Fing's Dynamics? 1, 2, or 3?	

ror (*JPos3D*) calculated as the normalized sum of estimated joints distances from the ground truth. Last is the Accuracy (*Acc*). We attend to highly constrained cases. Therefore, we indicate the amount of accuracy only if it is *not* 100%. The search algorithm knows the previous state of fingers in the sequence.

To animate the fingers, we employ a similar hierarchical *hand* database as proposed by (Dadgar and Brunnett, 2018). They Define their hierarchical database on various layers of complexity. That enables us to animate hand limbs individually with a specific emphasize on the layer of interest (e.g., fingers). Its finger's layer (e.g., L_4) is further partitionable into five sublayers (one for each finger) with modifiable step degree (e.g., resolution). These properties makes the database a suitable choice to examine the uniqueness of our *pose-descriptor* on different resolutions, refine it with different paths, and consider various variables to design specific scenarios.

We create a sequence of postures for each finger based on the temporal evolution of each finger specified for every experiment using a Viterbi-like algorithm (Viterbi, 1967). Returning $S = \{S_n | n = 1,2,3,4,5\}$ sequences (where $1 \equiv Little, 2 \equiv Ring, 3 \equiv Middle, 4 \equiv Index, 5 \equiv Thumb$) of $Q = \{q_1, q_2, ..., q_m\}$ states, where $m \approx 2000$ is a usual practice in this work. After selecting a specific global orientation, we retrieve the input sequences. Finally, the OpenCV's contour extraction method (Bradski, 2000) is applied to the input frames to extract the contours when searching for the optimal posture. All experiments employ the previous direction for the search. An elaborated version of the definitions and their evaluations are in the following subsections:

Experiment1. In this experiment, we consider all different digit combinations of fingers, including their transitions (see Fig8). The fingers start at closed-pose, and one by one (thus asynchronously), from the little finger, each of them opens and stays in the open pose (thus full-cycled on the finger level and having hand-level cycles) until all fingers open from right to left (therefore orderly or adjacent). After one-by-one closing, the next opening cycle starts from the next

Figure 8: A sample of inputs for experiment one.

Figure 9: A sample of inputs for experiment two.

(e.g., ring) finger. For this scenario, considering the $path_1$ where each finger has 21 possible poses from close to open, we have 2100 poses. For the first frame, we do not use the initialization (so the number of dynamics is 2^5). Nevertheless, for the end poses, at open or close cycles, the re-initialization information is known (thus, the number of possible dynamics for each finger is 1).

Experiment2. For this experiment, the collective free motion of fingers from close-posture to open (and reverse) is under investigation (see Fig9). Here, other fingers do not have to wait at their end states for one finger to reach its closed or opened states (thus there is no hand-level cycle). Identical to the previous experiment, the hand starts in the closed pose and not one-by-one (yet still asynchronously) opens from the little finger. Each finger opens till the end (thus full-cycled) until all fingers get opened from right to left (therefore adjacent). Considering the 21 poses of $path_1$, we have 1840 frames. Similarly, for the first frame we do not employ the initialization (thus the number of dynamics is 2^5). For the end poses, at each cycles, also, the re-initialization is known.

Experiment3. Here, we consider collective free motion (so there is no hand-level cycle) from close-posture to open (and reverse). Analogous to the previous experiment, the hand starts in the closed pose and asynchronously opens its fingers from the little one. However, unlike in two other experiments, each finger does not *fully* open until the end (thus partial-cycled) until all fingers reach their end-pose from right to left (therefore adjacent). For this scenario, considering the $path_1$, we have 2230 poses. We do not use the initialization for the first frame. However, the number of dynamics for that initial frame is 3^5 since the motion starts *not* at the beginning of the cycle. Here, for the end cycles, at open or close postures, the initialization information is available (thus, the number of possible dynamics for each finger is 1).

Evaluation. Using our *pose-descriptor* in the one-D temporal model enables us to achieve a real-time estimation, as shown in Table 2. In all experiments, average output frame rates are above 31 fps. These output

Table 2: The results of our three experiments indicate that our *pose-descriptor* can assist one to estimate the poses of all five fingers in real-time. The latency of the search is also within toleration if the frame rate of the input video is less than or equal to 31.

Exp		Descriptor $Type_1$			Descriptor $Type_2$		
	I_{fps}	O_{fps}	L_{Int}	L_{Ttl}	O_{fps}	L_{Int}	L_{Ttl}
1	25	32.5	62	142	32.5	50	156
2	25	32.4	52	173	31.1	44	302
3	24	31	440	515	32	294	359
1	32	32.5	632	1825	32.5	857	1791
2	31	32.4	237	971	31.1	1130	1734
3	31	31	2194	2202	32	1208	1474

frame rates are suitable to estimate the poses when the input videos have an fps of 25 or lower. Therefore, we can conclude our *pose-descriptor* provides an appropriate tool for real-time applications. The initialization of the first frame is the primary cause of the increase in the total latency (L_{Ttl}). The latency increases slightly during the rest of $\approx$ 2000 frames. That is usually the result of cost variations in contour extraction and Chamfer comparison caused by the alterable shape of the hand. However, because of much higher average output frame rates (compared to input fps), such increases in latency have marginal effects on the overall performance. The third experiment has a worse initial latency (L_{Init}). For, the third experiment's motions do not start at the full-cycled of close-postures. At those partial-cycled poses, there are three (previous, current, and the next) states for each finger to search ($3^5 = 243$). Whereas at full-cycled postures, there are solely two states (current and next). That leads to $2^5 = 32$ searches, and results in experiments one and two have better latency.

In rows four, five, and six of the table, we evaluate these experiments, this time with a higher input frame rate. As expected, the average output frame rates of the searches remain unchanged compared to the first set of experiments. However, the latency is different, and that affects the performance. Though the L_{Init} is worse, it is L_{Ttl} that experiences the highest diminution and affects the performance by large margins. When we increase the frame rates of the input video, the search algorithm finds the correct answer for each frame as before. However, a slight variation in the shapes of the contours and, thus, the estimation cost causes the search stays behind the video's current frames, and the L_{Ttl} drastically worsens. The primary purpose of the experiments was to demonstrate the suitability of our *descriptor* in estimating the finger poses in real-time applications, even when merely CPU resources are available. The conclusion section elaborates on the *JPose3D* and *Acc* metrics we achieved during the experimentation.

5 CONCLUSION

We proposed a simple *pose-descriptor* that characterizes the postures at finger level. We showed this descriptor has unique values for different finger poses, reducing the fingers' DoF to 5 and eliminates the necessity of constraining the problem (as needed in related works). We incorporated this *pose-descriptor* into a temporal model and with further modifications could achieve real-time performance on CPUs. To share more insights about the *JPose3D* and *Acc*, we briefly touch on other conducted experiments using the *GAT* paradigm, with various image scales and fingers' combinations.

To start systematically, we defined five categories of finger combinations: Cat_1 means merely the little finger is under the search. Cat_2 means solely the little and the ring fingers are the subjects of estimation. Cat_3 means we estimate the little, ring, and middle fingers pose. Cat_4 means we search the poses of the little, ring, middle, and index fingers. Finally, Cat_5 means we search all five fingers.

Beginning with *GAT* search and 100%-scale, we achieved $Acc = 100\%$ and $JPose3D = 0$, on Cat_1. As we proceeded to Cat_5, the results shows a slight decrements on the accuracy: $Cat_2(Acc = 100\%, JPose3D = 0)$, $Cat_3(Acc \approx 100\%, JPose3D = 0.0003)$, $Cat_4(Acc \approx 96\%, JPose3D = 0.001)$, $Cat_5(Acc \approx 95\%, JPos3D = 0.0008)$. However, the low average fps of the search was a significant obstacle since we were aiming for real-time results: $Cat_1(Out_{fps} = 1.23)$, $Cat_2(Out_{fps} = 0.41)$, $Cat_3(Out_{fps} = 0.14)$, $Cat_4(Out_{fps} = 0.045)$, and $Cat_5(Out_{fps} = 0.015)$.

Continuing with the *GAT* search, we down scaled the input and searched images. The lowest scale which led to fast yet accurate (stable) results was the 10%-scale. For example, on Cat_1 with that scale, we faced a slight decrease in accuracy $Cat_1(Acc = 97\%, JPose3D = 0.003)$. However, the average fps gain was considerable ($Out_{fps} = 25$). A similar pattern was observable in all categories insofar that for Cat_5, we achieved the accuracy of $Cat_5(Acc \approx 80\%, JPose3D = 0.02)$ and the average $(Out_{fps} = 0.3)$. Though having an acceptable accuracy, the time complexity for Cat_5 was still far from being real-time. However, the enhanced time complexity motivated us to modify the CD.

Proceeding to $GAT_{Dir_{1st}}$ search, we once considered various scales and coupled them with our modified Chamfer distance computation and reached the time complexity as high as $Cat_5(Out_{fps} = 42)$ on 10%-scale. However, without the scaling down, the average frame rate was around 32 for that specific experiment (not much difference). Thus we did not include scaling down the images in our experiment section to avoid the plethora of information.

Applications. The applications of our *pose-descriptor* can fork in several directions. First, as a consequence of reducing the finger's DoF, one could build a new motion capture system with fewer sensors (e.g., markers, haptics), wires, and circuitry. Second, our descriptor assists in constructing a training set that is as diverse as possible images in machine learning) to let the nets generalize better. Finally, one can benefit from our one-dimensional descriptor for studying and modeling of sign languages. We touched on this briefly and experienced the convenience of designing synthetic sign gestures with our descriptor.

ACKNOWLEDGMENTS

This project was funded by the Deutsche Forschungsgemeinschaft CRC 1410 / project ID 416228727.

REFERENCES

Bradski, G. (2000). The OpenCV Lib. *Dobb Sftwr Tl s Jrnl*.

Dadgar, A. and Brunnett, G. (2018). Multi-Forest Classification and Layered Exhaustive Search Using a Fully Hierarchical Hand Posture / Gesture Database. In *VISAPP*, Funchal.

Dung, L. and Mizukawa, M. (2010). Fast fingertips positioning based on distance-based feature pixels. *ICCE*.

Hardenberg, C. V. (2001). Bare-Hand Human-Computer Interaction.

Liao, Y., Zhou, Y., Zhou, H., and Liang, Z. (2012). Fingertips detection algorithm based on skin colour filtering and distance transformation. *Quality Sftwr Int Conf*.

Paralic, M. (2012). Fast connected component labeling in binary images. *35th Intern Conf on TSP*.

Raheja, J. L., Chaudhary, A., and Singal, K. (2011). Tracking of fingertips and centers of palm using KINECT. *CIMSim, 3rd Intern Conf*.

Shimizume, T., Umezawa, T., and Osawa, N. (2019). *Estimation of the Distance Between Fingertips Using Silhouette and Texture Information of Dorsal of Hand*, volume 11844 LNCS. Springer International.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans on Informtin Theory*, 13(2):260–269.

Yamamoto, S., Funahashi, K., and Iwahori, Y. (2012). A study for vision based data glove considering hidden fingertip with self-occlusion. *13th Int Conf on Software Eng, AI, Netwrk, & Parallel/Distrb Compt*.

Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., and Grundmann, M. (2020). MediaPipe Hands: On-device Real-time Hand Tracking.

Zhao, W., Zhang, J., Min, J., and Chai, J. (2013). Robust realtime physics-based motion control for human grasping. *ACM Trans on Graphics*, 32(6).

APPENDIX: UNIQUENESS

According to Fig3, we have $z_1 = L_1 \times cos(\theta_1)$, $z_2 = L_2 \times cos(\theta_2)$, $z_3 = L_3 \times cos(\theta_3)$, $z_4 = L_4 \times cos(\theta_4)$, $x_1 = L_1 \times sin(\theta_1)$, $x_2 = L_2 \times sin(\theta_2)$, $x_3 = L_3 \times sin(\theta_3)$, $x_4 = L_4 \times sin(\theta_4)$. Therefore, descriptor $Type_2$ will be:

$$\begin{aligned}D^2 &= F(\theta_1,\theta_2,\theta_3,\theta_4) = X^2 + Z^2 = (x_1+x_2+x_3+x_4)^2 + (z_1+z_2+z_3+z_4)^2 \\ &= L_1^2 sin^2(\theta_1) + L_2^2 sin^2(\theta_2) + L_3^2 sin^2(\theta_3) + L_4^2 sin^2(\theta_4) + \\ &L_1^2 cos^2(\theta_1) + L_2^2 cos^2(\theta_2) + L_3^2 cos^2(\theta_3) + L_4^2 cos^2(\theta_4) + 2 \times [\\ &L_1 L_2 sin(\theta_1) sin(\theta_2) + L_1 L_3 sin(\theta_1) sin(\theta_3) + L_1 L_4 sin(\theta_1) sin(\theta_4) + \\ &L_2 L_3 sin(\theta_2) sin(\theta_3) + L_2 L_4 sin(\theta_2) sin(\theta_4) + L_3 L_4 sin(\theta_3) sin(\theta_4) + \\ &L_1 L_2 cos(\theta_1) cos(\theta_2) + L_1 L_3 cos(\theta_1) cos(\theta_3) + L_1 L_4 cos(\theta_1) cos(\theta_4) + \\ &L_2 L_3 cos(\theta_2) cos(\theta_3) + L_2 L_4 cos(\theta_2) cos(\theta_4) + L_3 L_4 cos(\theta_3) cos(\theta_4)]\end{aligned}$$

Using $cos(x-y) = cosxcosy + sinxsiny$ and considering θ_1 is always zero, we can simplify the kinematic function $F(\theta_2,\theta_3,\theta_4)$ as following:

$$\begin{aligned}F &= L_1^2 + L_2^2 + L_3^2 + L_4^2 + 2 \times [L_1 L_2 cos(\theta_2) + L_1 L_3 cos(\theta_3) + \\ &L_1 L_4 cos(\theta_4) + L_2 L_3 cos(\theta_3 - \theta_2) + L_2 L_4 cos(\theta_4 - \theta_2) + L_3 L_4 cos(\theta_4 - \theta_3)]\end{aligned}$$

A Line Connecting Two Points. To show that $F(\theta)$ is unique/injective in a given interval (e.g., $(\theta_{2_0},\theta_{3_0},\theta_{4_0}) \neq (\theta_{2_1},\theta_{3_1},\theta_{4_1})$ with $F(\theta_{2_0},\theta_{3_0},\theta_{4_0}) \neq F(\theta_{2_1},\theta_{3_1},\theta_{4_1})$), we connect these two points with a line, and represent it in a vector form:

$$l(t) = \left(\theta_{2_0},\theta_{3_0},\theta_{4_0}\right) + t\left(\theta_{2_1}-\theta_{2_0},\theta_{3_1}-\theta_{3_0},\theta_{4_1}-\theta_{4_0}\right), \quad t \in \Re$$

In terms of $F(\theta)$, this line has the following form:

$$\overline{F}(t) = F\left(\theta_{2_0} + (\theta_{2_1}-\theta_{2_0})t,\ \theta_{3_0} + (\theta_{3_1}-\theta_{3_0})t,\ \theta_{4_0} + (\theta_{4_1}-\theta_{4_0})t\right)$$

With the following components:

$$\begin{aligned}\overline{F}_{\theta_2}(t) =& L_1^2 + L_2^2 + L_3^2 + L_4^2 + 2 \times L_1 L_2 cos(\theta_{2_0} + (\theta_{2_1} - \theta_{2_0})t) + \\ & 2 \times L_2 L_3 cos(\theta_{3_0} - \theta_{2_0} + (\theta_{3_1} - \theta_{2_1})t - (\theta_{3_0} - \theta_{2_0})t) + \\ & 2 \times L_2 L_4 cos(\theta_{4_0} - \theta_{2_0} + (\theta_{4_1} - \theta_{2_1})t - (\theta_{4_0} - \theta_{2_0})t) \\ \overline{F}_{\theta_3}(t) =& L_1^2 + L_2^2 + L_3^2 + L_4^2 + 2 \times L_1 L_3 cos(\theta_{3_0} + (\theta_{3_1} - \theta_{3_0})t) + \\ & 2 \times L_2 L_3 cos(\theta_{3_0} - \theta_{2_0} + (\theta_{3_1} - \theta_{2_1})t - (\theta_{3_0} - \theta_{2_0})t) + \\ & 2 \times L_3 L_4 cos(\theta_{4_0} - \theta_{3_0} + (\theta_{4_1} - \theta_{3_1})t - (\theta_{4_0} - \theta_{3_0})t) \\ \overline{F}_{\theta_4}(t) =& L_1^2 + L_2^2 + L_3^2 + L_4^2 + 2 \times L_1 L_4 cos(\theta_{4_0} + (\theta_{4_1} - \theta_{4_0})t) + \\ & 2 \times L_2 L_4 cos(\theta_{4_0} - \theta_{2_0} + (\theta_{4_1} - \theta_{2_1})t - (\theta_{4_0} - \theta_{2_0})t) + \\ & 2 \times L_3 L_4 cos(\theta_{4_0} - \theta_{3_0} + (\theta_{4_1} - \theta_{3_1})t - (\theta_{4_0} - \theta_{3_0})t)\end{aligned} \quad (6)$$

If the derivative of this function is non-zero for each t and any pair of points, then F is injective. We have

$$\overline{F}'(t) = (\theta_{2_1} - \theta_{2_0})\overline{F}_{\theta_2}(t) + (\theta_{3_1} - \theta_{3_0})\overline{F}_{\theta_3}(t) + (\theta_{4_1} - \theta_{4_0})\overline{F}_{\theta_4}(t) \quad (7)$$

Also, for human hands, one can realize the following:

$$\theta_2 = \boxed{[0,90^\circ]}, \theta_3 = \theta_2 + [0,120^\circ] = [0,210^\circ] \Longrightarrow \boxed{\theta_3 - \theta_2 = [0,120^\circ]}$$

$$\begin{aligned}\theta_4 =& \theta_3 + [0,45^\circ] = \theta_2 + [0,120^\circ] + [0,45^\circ] = \theta_2 + [0,165^\circ] = \\ & [0,255^\circ] \Longrightarrow \boxed{\theta_4 - \theta_2 = [0,165^\circ]}, \quad \boxed{\theta_4 - \theta_3 = [0,45^\circ]}\end{aligned}$$

Now, by numerically populating $Theta_i$s (combination of 2 from 486K, C^2_{486K}), L_is, and t, we check if the $F(\theta)$ is injective.