A geometric approach for learning compliant motions from demonstration

Markku Suomalainen and Ville Kyrki

Abstract— This paper proposes a method to learn from human demonstration compliant contact motions, which take advantage of interaction forces between workpieces to align them, even when contact force may occur from different directions on different instances of reproduction. To manage the uncertainty in unstructured conditions, the motions learned with our method can be reproduced with an impedance controller. Learning from Demonstration is used because the planning of compliant motions in 3-D is computationally intractable. The proposed method will learn an individual compliant motion, many of which can be combined to solve more complex tasks. The method is based on measuring simultaneously the direction of motion and the forces acting on the end-effector. From these measurements we construct a set of constraints for motion directions which, with correct compliance, result in the observed motion. Constraints from multiple demonstrations are projected into a 2-D angular coordinate system where their intersection is determined to find a set of feasible desired directions, of which a single motion direction is chosen. The work is based on the assumption that movement in directions other than the desired direction is caused by interaction forces. Using this assumption, we infer the number of compliant axes and, if required, their directions. Experiments with a KUKA LWR4+ show that our method can successfully reproduce motions which require taking advantage of the environment.

I. INTRODUCTION

Many assembly tasks require high precision. Humans manage such tasks through taking advantage of interaction forces caused by contacts between objects. These kind of motions are often called compliant motions. This paper looks into the question how compliant motions can be programmed easily. An example of compliant motion in workpiece alignment is depicted in Fig. 1.

To perform compliant motions, the interaction forces between objects must be limited. The simplest method for this is hybrid force-position control which, however, requires switching if the contact configuration changes and often exhibits force overshoot at contact transitions [1]. To alleviate this, it is beneficial to be able to use the same controller for both free space and contact motions. Impedance control is a natural choice for this [2]. An impedance controller features a virtual spring with adjustable stiffness, which allows a controlled amount of deviation from the planned trajectory.

Impedance control has the drawback that it makes motion planning difficult. Preimage planning [3] can be used for planning compliant motions in two dimensions. However, in

Fig. 1: Compliant motions can be used to align workpieces.

3-D preimage planning has been shown to be computationally infeasible [4]. There is an on-going interest in developing methods for motion planning under uncertainty [5].

In contrast to automatic planning, learning from demonstration (LfD) [6] can be used to transfer skills from a human expert to a robotic system. Most LfD methods try to reproduce any kinds of motions, and use tools such as dynamic movement primitives (DMP) [7] or hidden semi-Markov models [8]. Recently, there has been research to use DMP for assembly and workpiece alignment as well [9]. However, as DMP's use attractors to form the trajectory, they cannot take full advantage of geometries such as in Fig. 1, where contact force from either side could be used to guide the workpiece into the correct place.

We propose a method for learning compliant assembly motions from demonstrations. The intuition of our method stems from geometry: there is always a certain range of angles from which a human can push an object to make it slide along a surface. We use kinesthetic teaching to collect the demonstrations. From these demonstrations, both position and force data are recorded. From one or more recordings we learn the direction which will lead the end-effector through the motion, either directly in free-space or in contact. In addition, we learn the necessary compliant axes for performing the current task.

To allow the recording of forces in kinesthetic teaching, the robot must be equipped with a force/torque sensor and have a gravity compensation mode. The main application of our method is an assembly task consisting of phases and we assume that each phase can be defined by a motion with a single desired motion direction, possibly with directions of compliance to facilitate sliding in contact. Consequently, our method can be viewed as a primitive in a complex task. The combination of primitives for more complex tasks is outside the scope of this paper, but there are numerous publications on the topic (for example [10]).

The biggest advantage our method has over traditional LfD methods is the aforementioned ability to take advantage of the current task's geometry. With suitable workpiece design, this will greatly increase the the robustness of the method.

Force measurement is only required when collecting the demonstrations, and the sensor can be removed for reproduction since there is no force feedback. It is not necessary to know the exact location or orientation of the target workpiece due to the robustness against position errors, and therefore vision is not strictly required. This enables the use of our method in more difficult and unpredictable environments than, for example, DMP for workpiece alignment.

Our earlier work [11] considered a similar problem under the assumption that the forces exerted by human during demonstration can be directly measured, as is the case for example in teleoperation. In contrast, this work provides a solution to a more general problem where only the interaction forces can be measured.

Section II reviews related work in use of compliant motions and LfD. In Section III we explain the model used for learning the direction of motion and compliant axes in various scenarios. Next, experiments with a KUKA LWR4+ robot arm and their results are presented in Section IV. Finally, the results are discussed and future work is outlined in Section V.

II. RELATED WORK

The benefits of impedance control and compliant motions in assembly tasks have been known for a long time [12] and are still under active study [13]. However, most industrial robots sold today are still not equipped to perform impedance control [14]. In industrial settings, passive compliance [15] is often used to limit forces at time of initial contact. In contrast, this paper considers active compliance using impedance control in order to have adjustable compliance. Similar idea of motions constrained by physical restraints has been applied to manipulating articulated objects [16] by learning task dynamics. In contrast, in this paper we aim to identify a control strategy which is applicable over a variety of dynamics.

There are a few approaches for learning compliance parameters. Kronander and Billard [17] proposed learning them from human demonstration by halting a trajectory demonstration at intervals and using direct human demonstration of desired compliance by wiggling the robot. Later they developed a method for also increasing the compliance by gripping the robot tighter [18]. However, wiggling a tool which is in contact is not physically feasible and therefore the method is not suitable for in-contact tasks as considered in this paper. In addition, our proposed method estimates the parameters from demonstrations without extra halting.

Automatic learning of the stiffness matrix has been proposed by Carrera et al. [19] and Rozo et al. [20]. However, Carrera et al. used DMPs for the learning. We observed that DMPs cannot reproduce motions where at a stage of the motion there are multiple correct directions of motion, such as inserting a tool to a funnel. Rozo et al. learned the stiffness matrix from positional deviation using least squares estimation in a cooperative assembly scenario. The main difference between our method and Rozo et al. is that we require the interaction forces to complete the task, whereas in

Fig. 2: Force/torque sensor configuration, the position where external force by the human teacher $\boldsymbol{F_{ext}}$ is applied and the forces which sum up to the reading of the force/torque sensor.

their work the forces were an unnecessary consequence of the collaborative partner's actions. In our previous publication [11] we assumed that it is possible to directly measure the forces initiated by the human teacher, but in this paper we solve the more general case where only the interaction forces between the tool and the environment can be observed.

III. METHOD

We assume that a force/torque sensor is mounted between the location where the human holds the robot and the end-effector. One possible setup is illustrated in Fig. 2. The force/torque sensor measures the forces between the sensor and the contact point in a Cartesian coordinate system with compensation of the gravity force of the tool. Then when sliding the end-effector along a surface, the force measured by the force/torque sensor $\boldsymbol{F_m}$ is the combination of normal force of the surface $\boldsymbol{F_N}$, Coulomb friction working against the motion, and acceleration of the end-effector $\boldsymbol{a}$,

$$\boldsymbol{F_m} = \boldsymbol{F_N} + |\mu \boldsymbol{F_N}| (-\hat{\boldsymbol{v}}_{\boldsymbol{a}}) + m\boldsymbol{a} \tag{1}$$

where $\hat{\boldsymbol{v}}_{\boldsymbol{a}}$ is the direction of tool motion. Throughout this paper, we will use the circumflex notation to denote normalized version of a vector. The forces are defined in the world coordinate system and obtained via forward kinematics. We assume that the demonstrations are performed with almost constant speed, in which case we can ignore the acceleration term $m\boldsymbol{a}$.

Assuming the human demonstrator exerts a constant force and the system is in a stationary case where the velocity is constant, a particular motion direction $\hat{\boldsymbol{v}}_{\boldsymbol{a}}$ can result from any of a set of force directions presented as unit vectors $\hat{\boldsymbol{v}}_{\boldsymbol{d}}$, which we call the desired directions (see Fig. 2). The set of desired directions, $\hat{\boldsymbol{v}}_{\boldsymbol{d}}$, include all directions between $-\hat{\boldsymbol{F}}_{\boldsymbol{m}}$ and $\hat{\boldsymbol{v}}_{\boldsymbol{a}}$, because a force applied along any of those would cause sliding in the direction $\hat{\boldsymbol{v}}_{\boldsymbol{a}}$.

Before considering how $\hat{\boldsymbol{v}}_{\boldsymbol{d}}$ can be estimated from demonstrations, we note that we assume that a complex task can be divided into simple motion segments, each of which has a single desired direction of motion. Applying force to end-effector in this desired direction will bring it to the desired position of the current motion, either by free-space motion or sliding along a surface leading towards the position. This paper does not consider the problem of

(a) Human variation in 2-D   (b) P in 3-D

Fig. 3: Visualizations of how (a) the error ϵ from (2) is added to both $\hat{F}$ and $\hat{v}_a$ and (b) how the polyhedron P from (3) is constructed at each time step.

dividing a demonstration into segments and we assume that the segmentation can be performed.

A. Learning desired direction

We define the desired position of a motion as a point or a set of points where the demonstrator wants the end-effector to move to. The desired direction of motion $\hat{v}_d^*$, chosen from the set of possible desired directions $\hat{v}_d$, is a direction vector in 3-D Cartesian space. If the robot applies a force in this direction, the end-effector will move towards the desired position, either by free-space motion or by sliding along a surface. The whole process of finding $\hat{v}_d^*$ is formulated in Algorithm 1.

From Fig. 2 and (1) we observe that in the direction of motion, $\hat{v}_d$ is constrained by $\hat{v}_a$ and $-\hat{F}_m$. In addition, we consider that a human can not demonstrate a desired trajectory perfectly, resulting in small differences between the demonstrated $\hat{v}_a$ and human intent. Because contact constrains the motion, this needs to be taken into account only perpendicular to $\hat{v}_a$. We define the maximum error ϵ such that it forms an angle of α degrees with the plane spanned by unit vectors $\hat{v}_a$ and $\hat{F} \equiv -\hat{F}_m$, as illustrated in Fig. 3a. Formally, we write

$$\epsilon = \tan\alpha \frac{\hat{F} \times \hat{v}_a}{|\hat{F} \times \hat{v}_a|} \tag{2}$$

Now for each point k along the demonstrated trajectory, we can construct a polyhedron P consisting of the following vectors commencing from origin measured at point k:

$$P_k = \begin{pmatrix} \hat{v}_a + \epsilon \\ \hat{v}_a - \epsilon \\ \hat{F} - \epsilon \\ \hat{F} + \epsilon \end{pmatrix} \tag{3}$$

The polyhedron is illustrated in Fig. 3b. P represents the set of possible desired directions of motion $\hat{v}_d$ of a single point in 3-D. This is described in Algorithm 1 on row 2.

A small value of measured force F_m signifies free space motion and renders the result of normalization in (2) volatile. We detect such cases using a manually chosen threshold for F_m. When the force threshold is not met, we construct the polyhedron P from a number of points on a circle with

---

Algorithm 1 The whole algorithm to calculate desired direction from demonstrations

1: Find R which rotates $\bar{v}_a$ to positive z axis
2: **for** each point k **do**
3:     Calculate P from (2) and (3)
4:     Rotate P: $P_r = RP$
5:     Call Algorithm 2 for each p in P_r to compute Θ
6: **end for**
7: $G(i,j) = 0 \; \forall \; i,j$
8: **for** Each cell (i,j) in G **do**
9:     **for** Each Θ **do**
10:         **if** (i,j) inside Θ **then**
11:             $G(i,j) = G(i,j) + 1$
12:         **end if**
13:     **end for**
14: **end for**
15: $g_{i,j}$ indices of vector median of $\max(G)$
16: **for** each Θ **do**
17:     **if** $g_{i,j}$ inside Θ **then**
18:         add Θ to Θ^*
19:     **end if**
20: **end for**
21: Compute intersection Φ of all Θ^*
22: Calculate Chebyshev center ϕ^* of Φ
23: Call Algorithm 3 with ϕ^* to compute $\hat{v}_{des-r}^*$
24: Rotate back: $\hat{v}_d^* = R^{-1}\hat{v}_{des-r}^*$

---

center at $\hat{v}_a$, perpendicular to it, and with radius ϵ. With this addition, our method covers pure free space motion as well.

To find the set of feasible desired directions for the whole motion, we must satisfy all the constraints through the trajectory represented by n polyhedra P. To simplify the computation, we transform the direction vectors to a 2-D angular space. Thus we project the 3-dimensional polyhedra into 2-dimensional polygons. This is performed by calling Algorithm 2 with polyhedra P.

Algorithm 2 Conversion from Cartesian 3-D vector p to angular 2-D vector Θ.

1: Normalize p
2: $r = \arccos(p_z)$
3: $\gamma = \arctan 2(p_y, p_x)$
4: $\theta_x = r\cos(\gamma)$
5: $\theta_y = r\sin(\gamma)$

To avoid computational problems going from $-\pi$ to π radians in the angular coordinate system, we rotate the vectors before conversion such that the resulting angles are centered around zero. To do this, we find a rotation matrix R such that it rotates the mean direction of motion of a demonstration $\bar{v}_a$ to match with the positive z axis using Rodrigues formula. After performing this on row 3 in Algorithm 1, we can call Algorithm 2 with the result. Then we form a rectangle Θ from the four (θ_x, θ_y) pairs

calculated with Algorithm 2 describing the constraints in the angle coordinate system, as on row 4 in Algorithm 1. These points are the angular coordinates of polyhedron P.

To find an intersection of these n rectangles, outliers must be rejected. We address this by determining inliers by voting. First we construct a voting grid, matrix G of zeros with a chosen resolution, for example 1 degree, as on row 6 in Algorithm 1. The indices of this matrix relate to values in the angular coordinate system. Then for each element in G, we check if the element, according to its indices, is inside each polygon Θ. If it is, we add 1 to the value of the element. These are rows 7–13 in Algorithm 1. The result is a matrix G with one or more maxima. An example calculated from two perpendicular funnel demonstrations is illustrated in Fig. 4.

Fig. 4: Heatmap illustration of the voting grid depicting the polygon intersection used for outlier rejection. The colormap unit describes the number of Θ within which each pixel lies.

If there are more than one maximum values, we choose their vector median [21] g_{ij}, as described on rows 14–15 in Algorithm 1. We then loop over polygons Θ again and calculate the intersection of the set of polygons Θ^* in which the indices of g_{ij} lie, as shown on lines 16–20 in Algorithm 1. We call the intersection polygon Φ. It describes the set of feasible desired directions $\hat{v}_d$ in the angular coordinate system. To find a single direction, $\hat{v}_d^*$, we need to choose the corresponding angular system point ϕ^* from Φ. We choose ϕ^* as the Chebyshev center of Φ, i.e. the center of largest circle inscribed in the polygon [22].

Finally, we need to convert ϕ^* to a 3-dimensional vector $\hat{v}_{des-r}^*$. This occurs on row 23 in Algorithm 1 and is presented in detail in Algorithm 3. The variable s is needed to deduce the correct sign for the resulting vector, since $\arctan 2$ can produce this information whereas the ratio of $\hat{p_x}$ and $\hat{p_y}$ cannot. The result is a unit vector, but only directions are important in our application.

We still need to rotate $\hat{v}_{des-r}^*$ back to the original coordinate system, by calculating $\hat{v}_d^* = R^{-1}\hat{v}_{des-r}^*$. Now the result is the direction along which a force must be applied to guide the end-effector to the desired position.

If we have multiple demonstrations, the method works exactly the same way. The polygons Θ are concatenated and an intersection is calculated. Difference between demonstrations simply makes Φ smaller. If there are multiple viable approach directions, all of them should be demonstrated to

Algorithm 3 Conversion from angular 2-D vector Θ to Cartesian 3-D unit vector $\hat{p}$.

1: $s = \text{sign}(\arctan 2(\theta_y, \theta_x))$
2: $r = \cos\left(\sqrt{\theta_x^2 + \theta_y^2}\right)$
3: $a = \frac{\theta_y}{\theta_x}$
4: $\hat{p}_z = \arccos(r)$
5: $\hat{p}_x = s\sqrt{\frac{1-\hat{p}_z^2}{1+a^2}}$
6: $\hat{p}_y = s \cdot a \cdot \hat{p}_x$

allow maximal utilization of environment in reproduction. For example, in the case of the funnel, the demonstrations should come from parallel directions for the method to learn to slide along any side. If two demonstration from nearly same directions are given, the method will assume that this is the only possible direction to approach from and can only replicate such motions. No more than one demonstration from each approach direction is required if the demonstration is well performed.

B. Learning axes of compliance

To successfully complete a motion, the compliant axes need to be identified. If the motion occurs only in free space, no compliance is required. In-contact tasks require at least one compliant axis. Certain tasks, such as inserting a tool to a funnel, require a second compliant axis to take full advantage of the funnel's geometry and allow sliding along any side of the funnel. We assume that if compliance is required, the axis should be totally compliant.

We first observe that the compliant axes must be perpendicular to the desired direction $\hat{v}_d^*$. This is due to our definition of no stiffness along a compliant axis—the end-effector would not move in that direction even if commanded. The key idea is that if there is movement along other directions besides the desired direction, this movement must be generated by interaction forces. When the environment interaction is causing forces on the end-effector, compliance is needed to reproduce the demonstrated motions. In contrast, if only force but no motion is measured in a certain direction, this direction must be stiff to avoid instability.

To exploit this idea, we first calculate the mean direction of actual motion $\bar{v}_a$ separately for each demonstration used in calculation of corresponding $\hat{v}_d^*$. Then we rotate all $\bar{v}_a$ such that in the new coordinate system $\hat{v}_d^*$ is along the z-axis, after which we use Algorithm 2 to convert the set of $\bar{v}_a$ to angular coordinates ϕ_a. As a result, the values ϕ_a will be within a unit circle, where origin represents $\hat{v}_d^*$ and the values on the unit circle are directions perpendicular to $\hat{v}_d^*$. As we assume compliance is required in the direction where we observe motion without human initiative, we can see that the direction of compliance should be towards ϕ_a and lie on the unit circle to fulfil the orthogonality requirement. This is illustrated in Fig. 5.

To choose the correct number of compliant axes, we need to measure how well each model (number of compliant axes) explains the observations, but also discourage the choice

| (a) 2 compliant axes | (b) 1 compliant axis |

Fig. 5: Unit spheres in the coordinate system for determining the compliant axes. Origin (small black circle) represents $\hat{v}_d^*$ in the angular coordinates, green crosses are the corresponding ϕ_a of each demonstration, blue line is the linear model u and red cross the identified compliant direction when the model with 1 compliant axis is chosen.

of an overly complicated model. We take our inspiration from the Bayesian Information Criterion (BIC) [23], which is defined

$$BIC = \ln(n)k - 2\ln(L) \tag{4}$$

where n is the number of data points, k the number of parameters and L the likelihood of a model.

We assume that the error a human makes while demonstrating a task is normally distributed with variance σ. Therefore we calculate the likelihoods for each model from a 2-D normal distribution

$$L_i = \prod_j \mathcal{N}\left(\epsilon_{i,j} \mid \mathbf{0}, \begin{pmatrix} \sigma & 0 \\ 0 & \sigma \end{pmatrix}\right). \tag{5}$$

where $\epsilon_{i,j}$ is the error of data point j in model i.

For the case of no compliant axes (i.e. free space motion) under no errors $\phi_a = \mathbf{0}$. Therefore, the error is defined $\epsilon_0 = \phi_a$. With a single compliant axis, the error is the distance from ϕ_a to a line u, illustrated in Fig. 5, which maximizes the likelihood of ϕ_a and passes through the origin. Because normal distribution is isotropic, we can set $\epsilon_1 = [u^T \phi_a \ 0]$. Finally, 2 compliant axes perfectly explain any data as the model describes the linear combination of two lines, and therefore we set $\epsilon_2 = [0 \ 0]$.

We calculate the BIC for each of the models and choose the one with lowest BIC. If the model with a single compliant axis is chosen, we choose the direction of compliance as the intersection between u and the unit circle, marked with red x in Fig 5. If no compliance is needed, a position controller is sufficient. If two compliant axes are needed, any pair of vectors which form an orthonormal base with $\hat{v}_d^*$ are adequate as the compliant axes. The whole process is summarized in Algorithm 4.

It should be noted that the proposed approach does not follow the typical use of BIC which is only applicable when $n \gg k$ and the variance in the likelihood is calculated from the data. Instead, we assume that the uncertainty

of demonstrations can be estimated beforehand, making it possible to use the proposed formulation.

Algorithm 4 Finding the required number of compliant axes and their directions

1: Find R such that $R\hat{v}_d = \hat{z}$
2: Calculate set of mean actual directions $\bar{v}_a$
3: Rotate $\bar{v}_{ar} = R\bar{v}_a$
4: Apply Algorithm 2 to convert $\bar{v}_{ar}$ to angular coordinates ϕ_a
5: Calculate $L_0 = \mathcal{N}(\phi_a | \mathbf{0}, \Sigma)$
6: Calculate $L_1 = \mathcal{N}(\epsilon_1 | \mathbf{0}, \Sigma)$, where $\epsilon_1 = [u^T \phi_a \ 0]$
7: Calculate $L_2 = \mathcal{N}(\mathbf{0} | \mathbf{0}, \Sigma)$
8: Calculate BIC (4) of L_0, L_1 and L_2. Choose the model with lowest BIC value.

C. Reproduction

An impedance controller is used for the reproduction. It is a feedback controller defined as

$$F = K(x^* - x) + Dv + f_{dyn}, \tag{6}$$

where x^* is the desired position, x the current position, K a gain matrix, Dv a linear damping term and f_{dyn} the feed-forward dynamics of the robot including gravity. The reproduction is performed similarly as in [11]. The stiffness is set according to $K = RVR^T$, where V is a diagonal matrix defining the stiffness and the number of compliant axes. R defines the directions of compliant axes and is constructed from $\hat{v}_d^*$ and the compliant axes discovered with Algorithm 4. The desired positions for each time step are calculated in a feed-forward manner from $\hat{v}_d^*$ and the desired velocity v as

$$x_t^* = x_{t-1}^* + \hat{v}_d^* v \Delta t \tag{7}$$

where Δt the sample time of the control loop.

IV. EXPERIMENTS AND RESULTS

The robot used for the experiments was a KUKA LWR4+ lightweight arm. The control law for the Cartesian impedance control of the KUKA LWR4+ through KUKA Fast Research Interface (FRI) [24] is

$$\tau = J^T(\text{diag}(k_{FRI})(x^* - x) + \text{diag}(d_{FRI})v + F_{FRI}) + f_{dyn} \tag{8}$$

where $\text{diag}(k_{FRI})$ is a diagonal matrix constructed of the gain values of k_{FRI}. As in [11], we implemented our controller through the F_{FRI} by setting $k_{FRI} = 0$ and $F_{FRI} = K(x^* - x)$, getting a controller equal to Eq. (6) where f_{dyn} is managed by the KUKA's internal controller. We set the damping value through d_{FRI} to $0.7 \frac{N \cdot s}{m}$, the stiffness of the compliant axes through our own controller to 0 and the non-compliant axes to $500 \frac{N}{m}$ for safety reasons.

To test our method, we built two different test setups. First one is a valley consisting of two aluminium plates placed in 90 degrees angle with each other. The second one is a plastic funnel with curved slopes. In addition, we tested the

(a) Robot and valley   (b) Funnels

Fig. 6: Our test equipment, the KUKA LWR4+ lightweight arm, the valley setup, and the two funnels, curved and straight.

reproduction on another funnel, which had straight slopes. All of the hardware we used is presented in Fig. 6.

In practice, due to noise in the demonstration from human and measurement uncertainty, averaging over a chosen number of time steps to compute P of (3) produces more stable results. To filter the noise, we chose to average over 20 time steps of original 100Hz measuring frequency, which meant sampling P in 5Hz. We used a manually estimated maximum error angle of 20 degrees for human demonstration for value of α in (2).

A. Desired motion direction

To validate that we can learn the desired direction from both curved and straight surfaces and multiple directions, we executed a number of experiments on the physical setups presented in Fig. 6. We also wanted to study the number of demonstrations required for learning $\hat{v}_d^*$. Unfortunately, no alternative methods exist in the literature that could be used for comparison, the closest alternative being our earlier work [11] which assumes direct measurement of demonstrator forces.

To better illustrate the process, we drew the polyhedra P from (3) at intervals in 3-D in Fig. 7. The trajectory is from a funnel setup, which explains the curved shape. The form of Fig. 3b is distinguishable and it can be seen how the direction of P changes over time.

Fig. 7: The polyhedra P drawn to a trajectory on a funnel demonstration. Units are in m.

(a) Valley setup   (b) Funnel setup

Fig. 8: 2-D polygons Θ (the axes representing θ_x and θ_y) of two demonstrations: (a) down different sides of the valley and (b) perpendicularly into the funnel. The red and blue rectangles represent the Θ of separate demonstrations, and the black polygon is the set of desired directions in angular coordinate system, Φ.

The intersection calculations are performed in 2-D on the polygons Θ calculated in Algorithm 2 and concatenated from multiple demonstrations. Figure 8a shows two trajectory demonstrations performed by sliding along different sides of the valley. The difference in the orientation of the polygons is due to the normal forces being in opposite directions, therefore constraining the final intersection Φ. Figure 8b shows two perpendicular funnel demonstrations. Now we can see that Θ are almost perpendicular. The funnel demonstrations were briefer than the valley, which explains that there are less Θ in Fig. 8b, but still the algorithm finds a clear intersection Φ.

We also wanted to verify our assumption that one demonstration from each viable direction is enough to calculate $\hat{v}_d^*$. To do this, we performed 32 demonstrations of the funnel motion, such that all the demonstrations were either perpendicular or opposite to each other. In this case every 4th demonstration came from the same direction. We assumed that in the funnel demonstration, the ground truth of $\hat{v}_d^*$ is directly downwards when the funnel is upright. Then we divided the demonstrations into groups of 2, 4, 8 and 16 demonstrations, calculated the $\hat{v}_d^*$ and it's error. Box plot of the results is in Fig. 9. We can see that the error was below our assumption of human error already on 2 demonstrations and did not decrease when more demonstrations were added. Therefore we conclude that one demonstration along each possible trajectory is enough to learn a valid $\hat{v}_d^*$.

B. Degrees of freedom

To verify our method for finding the degrees of freedom, we performed 30 demonstrations of free space motion with a straight downward trajectory and 30 demonstrations of sliding down the valley. In free space motion zero compliant degrees of freedom are required. Sliding down the valley requires one compliant degree of freedom, since sliding is required but only in a single direction. The funnel demonstration must be reproducible from anywhere within the projection of the funnel, and therefore two compliant degrees of freedom are required. The resulting BIC values can be seen in Fig. 10a for free space motion, Fig. 10b for the

Fig. 9: Box plot of the error angle with different number of demonstrations used to calculate $\hat{v}_d^*$. The edges of the blue boxes are the 25th and 75th percentiles of the data.

(a) BIC values for free space motion

(b) BIC values for the task of sliding to the bottom of the valley

(c) BIC values for insertion to funnel

Fig. 10: BIC values for three different experiments

demonstrations down the valley and Fig. 10c for the funnel. We can see that our algorithm performs the classification correctly. The choice of σ in (5) is important for these results. We used the value $\sigma = 0.03$, which corresponds to standard error deviation of approximately 10 degrees in the human demonstration.

To study the learning of a challenging motion, we performed 30 demonstrations of sliding along the side of the valley. Unlike the down the valley motion, which in our setup would work even with 2 degrees of freedom, strictly one degree of freedom aligned vertically was required for a successful reproduction. Figure 11 shows two combinations of two demonstrations in the same coordinate system as in Fig. 5. In Fig. 11a the actual directions v_a of the demonstrations are well aligned and the model with 1 compliant axis is correctly identified. However in Fig. 11b the demonstrations are not well aligned even if the human demonstrator attempted to demonstrate the same direction both times and therefore our algorithm detects that the model

(a) Valid demonstration   (b) Poor demonstration

Fig. 11: Two demonstrations of sliding along the side of the valley depicted on each figure. Same coordinate system as in Fig. 5.

with 2 compliant axes better explains the data. Therefore care must be taken when the demonstrations are given, especially if the minimum number of demonstrations is used for the algorithm and the motion is difficult for the human teacher to demonstrate.

C. Reproduction

We studied the execution robustness of our method by varying the starting positions of the motion and also by making changes in the test setups. First our algorithm learned $\hat{v}_d^*$ and the model with two compliant axes using 2 perpendicular demonstrations with the curved funnel. Figure 12 presents four different starting positions and setups in which the motion was successfully reproduced with the learned parameters. In Figs. 12a and 12b we used the same funnel as for learning with a varying starting position. In Figs. 12c and 12d we used the funnel with direct sides and tilted it 15 degrees. In addition, we performed successful reproductions in both valley setups after two demonstrations, sliding down the valley and along the side of the valley. We conclude that our method can successfully learn the required parameters from a small number of demonstrations to perform motions which take advantage of the environment.

V. CONCLUSIONS AND FUTURE WORK

We showed that our method can successfully learn to replicate assembly motions which require interaction with the environment. A user will need to give one or more kinesthetic demonstrations of the task using a robot with a force/torque sensor attached between the end-effector and the user's grasp. The motion is fully described by the desired direction, number of compliant axes and their directions. The desired direction is learned using the geometric intuition that a sliding motion is executed with a sufficient force in any direction between the direction of motion and the sum of normal force and Coulomb friction. The compliant axes are learned by assuming that they must be perpendicular to the desired direction and that all motion in other directions besides desired direction must be caused by compliance.

The main advantage of the method compared to existing LfD methods is that the method can learn how to take

(a)　　　　　　　(b)　　　　　　　(c)　　　　　　　(d)

Fig. 12: Possible setups in which the end-effector is successfully moved to the target at the bottom of the funnel.

advantage of physical guides to align workpieces. Since most assembly tasks can be represented as a sequence of motions, learning a single direction together with the compliant axes is sufficient. However, for tasks where contact force modulation is required, other methods are more suitable.

Our method does not solve a whole assembly task. The demonstration of a whole task must first be divided into motion segments which can be learned with our method. The segmentation could be performed using threshold values [13] or machine learning methods such as Hidden Markov Models [10]. The suitability of various segmentation approaches with our method remains a future study.

Many assembly tasks require rotational motions such as screwing. Therefore a natural extension for the method would be to learn rotational motions from demonstration. This has, to our knowledge, not been done before and we will pursue it in our future research.

REFERENCES

[1] O. Alkkiomaki, V. Kyrki, H. Kalviainen, Y. Liu, and H. Handroos, "Smooth transition from motion to force control in robotic mani pulation using vision," in *International Conference on Control, Automation, Robotics and Vision*, (Singapore), pp. 2037–2042, 12 2006.

[2] N. Hogan, "Stable execution of contact tasks using impedance control," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4, pp. 1047–1054, IEEE, 1987.

[3] T. Lozano-Perez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robots," *The International Journal of Robotics Research*, vol. 3, no. 1, pp. 3–24, 1984.

[4] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Foundations of Computer Science, 1987., 28th Annual Symposium on*, pp. 49–60, IEEE, 1987.

[5] M. Koval, N. Pollard, and S. Srinivasa, "Pre- and post-contact policy decomposition for planar contact manipulation under uncertainty," *The International Journal of Robotics Research*, vol. 35, no. 1–3, pp. 244–264, 2016.

[6] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[7] S. Schaal, "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics," in *Adaptive Motion of Animals and Machines*, pp. 261–280, Springer, 2006.

[8] M. Racca, J. Pajarinen, A. Montebelli, and V. Kyrki, "Learning in-contact control strategies from demonstration," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 688–695, IEEE, 2016.

[9] L. Peternel, T. Petrič, and J. Babič, "Human-in-the-loop approach for teaching robot assembly tasks using impedance control interface," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 1497–1502, IEEE, 2015.

[10] O. Kroemer, H. Van Hoof, G. Neumann, and J. Peters, "Learning to predict phases of manipulation tasks as hidden states," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 4009–4014, IEEE, 2014.

[11] M. Suomalainen and V. Kyrki, "Learning compliant assembly motions from demonstration," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 871–876, IEEE, 2016.

[12] W. Wang, R. N. Loh, and E. Y. Gu, "Passive compliance versus active compliance in robot-based automated assembly systems," *Industrial Robot: An International Journal*, vol. 25, no. 1, pp. 48–57, 1998.

[13] A. Stolt, *On Robotic Assembly using Contact Force Control and Estimation*. PhD thesis, Lund University, 2015.

[14] H. Chen and Y. Liu, "Robotic assembly automation using robust compliant control," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 2, pp. 293–300, 2013.

[15] S.-k. Yun, "Compliant manipulation for peg-in-hole: Is passive compliance a key to learn contact motion?," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 1647–1652, IEEE, 2008.

[16] T. Petrič, A. Gams, L. Žlajpah, and A. Ude, "Online learning of task-specific dynamics for periodic tasks," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 1790–1795, IEEE, 2014.

[17] K. Kronander and A. Billard, "Online learning of varying stiffness through physical human-robot interaction," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 1842–1849, Ieee, 2012.

[18] K. Kronander and A. Billard, "Learning compliant manipulation through kinesthetic and tactile human-robot interaction," *Haptics, IEEE Transactions on*, vol. 7, pp. 367–380, July 2014.

[19] A. Carrera, N. Palomeras, N. Hurtós, P. Kormushev, and M. Carreras, "Learning multiple strategies to perform a valve turning with underwater currents using an i-auv," in *OCEANS 2015-Genova*, pp. 1–8, IEEE, 2015.

[20] L. Rozo Castañeda, S. Calinon, D. Caldwell, P. Jimenez Schlegl, and C. Torras, "Learning collaborative impedance-based robot behaviors," in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, pp. 1422–1428, 2013.

[21] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proceedings of the IEEE*, vol. 78, no. 4, pp. 678–689, 1990.

[22] A. L. Garkavi, "On the Chebyshev center and convex hull of a set," *Uspekhi Matematicheskikh Nauk*, vol. 19, no. 6, pp. 139–145, 1964.

[23] G. Schwarz *et al.*, "Estimating the dimension of a model," *The annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[24] G. Schreiber, A. Stemmer, and R. Bischoff, "The fast research interface for the KUKA lightweight robot," in *Proc. of the IEEE Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications – How to Modify and Enhance Commercial Controllers (ICRA 2010)*, IEEE, 2010.

[25] A. L. P. Ureche, K. Umezawa, Y. Nakamura, and A. Billard, "Task parameterization using continuous constraints extracted from human demonstrations," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1458–1471, 2015.