Using hand synergies as an optimality criterion for planning human-like motions for mechanical hands

Jan Rosell and Raúl Suárez

***Abstract*— The use of anthropomorphic mechanical hands allows the execution of complex manipulation tasks and, therefore, its use is increasing, both in humanoid robots and mobile manipulators. The planning of the motions of these hands is not easy due to the high number of degrees of freedom (DOF) involved. Even when they may be mechanically independent, these degrees of freedom can be artificially coupled in order to mimic the human hand motions by using synergies, which in its turn results in a lower subspace where to perform the motion planning, as previously done by the authors using a PRM. This paper extends this idea by using an RRT*, that differently from other sampling-based planners, is an asymptotically optimal method. The optimization function selected evaluates the alignment of the path to the directions defined by the synergies, thus favoring human-like motions. The proposal is conceptually illustrated with a simple 2 DOF planar robot and applied to a 13 DOF mechanical hand.**

I. INTRODUCTION

The planning of motions for robotic systems with a high number of degrees of freedom (DOF), like those involving mechanical hands, is a challenge that can be tackled using sampling-based approaches like Probabilistic RoadMaps (PRM [1]) or Rapidly-exploring Random Trees (RRT [2]). These planners provide probabilistic completeness and work well in practice, and, moreover, several variants have been proposed in order to improve their performance in difficult problems, for instance, some of them use different importance sampling methods for PRMs to obtain samples in difficult regions of the configuration space like narrow passages [3], [4], while others propose techniques for RRTs to project samples onto the manifolds that contain the solutions to problems involving task manipulation constraints [5], [6]. All these algorithms, however, return non-optimal solutions. To settle this problem, new algorithms called PRM* and RRT* have been proposed, returning solutions that converge almost surely to the optimum [7].

The basic idea of RRTs is to build a tree of feasible motions, rooted at the initial configuration, by iteratively sampling a random configuration ($\boldsymbol{q}_{\text{rand}}$), searching the node of the tree nearest to it ($\boldsymbol{q}_{\text{near}}$), and growing an small amount from $\boldsymbol{q}_{\text{near}}$, in the direction of $\boldsymbol{q}_{\text{rand}}$, towards a new configuration $\boldsymbol{q}_{\text{new}}$. If the path connecting $\boldsymbol{q}_{\text{near}}$ and $\boldsymbol{q}_{\text{new}}$ is collision-free then it is added as an edge of the tree. In the RRT* algorithm, once $\boldsymbol{q}_{\text{new}}$ has been computed as in the RRT case, it is not directly connected to $\boldsymbol{q}_{\text{near}}$ but to the node (among a given set of neighbors) that minimizes the cost to reach $\boldsymbol{q}_{\text{new}}$. Then, RRT* checks whether each neighbor node can be reached, through $\boldsymbol{q}_{\text{new}}$, with a cost smaller than its current one and, if so, rewires the edges of the tree. Thanks to this rewiring process the solution keeps improving while the number of samples increases. For high-dimensional configuration spaces, however, RRT* may have a slow cost improvement due to the the heavy exploration nature of RRTs that makes it difficult for nodes of an already found solution path to be selected for the expansion and (possibly) rewired. To mitigate this effect some sampling biases have been proposed, like a Voronoi bias in the task space [8], a local bias around the solution path [9], or a bias based on projections learned during the planning process [10].

The authors are with the Institute of Industrial and Control Engineering (IOC), Universitat Politècnica de Catalunya (UPC) – Barcelona Tech, Barcelona, Spain, {jan.rosell, raul.suarez}@upc.edu. This work was partially supported by the Spanish Government through the projects DPI2010-15446, DPI2011-22471 and DPI2013-40882-P.

Regarding the problems related to mechanical hands, for those that are anthropomorphic, the use of Principal Component Analysis (PCA) has been widely used in order to mimic the human hand, because PCA applied to a set of samples of human hand configurations allows to capture the couplings that there exist in the human hand between the finger joints. When these couplings, called synergies, are mapped to the mechanical hand, human-like postures can be obtained. This approach has been used in the grasp synthesis problem [11], [12], in manipulation [13], in teleoperation [14], and in motion planning [15], where synergies were called Principal Motion Directions or PMDs (in this work either synergies or PMDs will be used indistinctively). In motion planning, besides the obtention of human-like motions, the use of PCA is useful to improve the planning performance due to the reduction of DOF obtained if only few synergies with a high accumulated variance are used. PCA has further been exploited as an importance sampling method either using PRMs or RRTs [16], [17].

This paper proposes the use of an RRT* and the use of hand synergies to obtain human-like motions in an optimal way. Since the first PMDs or synergies capture the main couplings of the hand, the proposal is to define an optimization function, to be used with the RRT* algorithm, that makes the tree edges to be as much aligned as possible with the vectors describing these synergies. That is, the more aligned to these synergies an edge is the lower its cost, and since the RRT* returns a solution that converges almost surely to the optimum, then the solution is optimal in the sense of human-likeness (considering as human-like movements those movements that follow the human synergies). A related approach was explored in [18], where an RRT* called

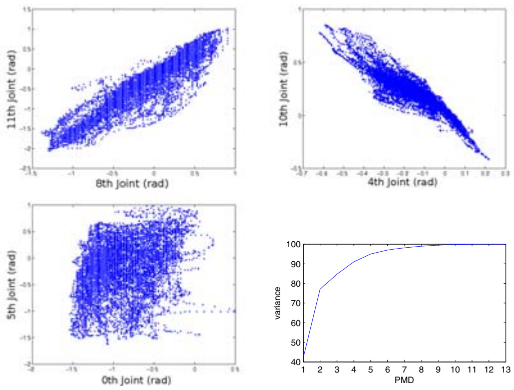

Fig. 1. Top-left: Positive correlation between proximal phalanxes; Top-right: Negative correlation between the index and the ring abductions/adductions; Bottom-left: No correlation between the thumb base and the medium phalanx of the index; Bottom-right: Total variance covered as a function of the number of PMDs used. Reproduced from [15].

coupled-RRT*, or cRRT*, was used to plan motions for a team of mobile robots in such a way that they moved like a flock of birds. This was achieved using the directions that determine the coupling between the translational DOF of each robot and an optimization function that forced the solution to be as much aligned as possible with them.

After this introduction, the paper is structured as follows. Section II explains how the hand synergies are obtained, Section III describes the proposed optimization function, and Section IV the planning procedure that is illustrated in Section V with two examples. Finally, SectionVI summarizes and discusses the proposal.

II. Hand synergies

The mapping of human hand synergies to the robotic hand can be done in different ways (see [19] for a good discussion on the topic). In our work, in order to plan human-like motions for a mechanical hand, the human hand synergies are transferred to the mechanical hand with the following procedure. First, using a sensorized glove (Cyberglove) the data from the joints of the human hand are captured while the operator is moving freely the fingers (i.e. without imposing actions related to grasping or manipulation of objects, which is a key difference with most previous works), and mapped to the mechanical hand Schunk SAH [20] using a joint-to-joint mapping. Then, the Principal Component Analysis is performed to the mapped data and the resulting eigenvectors and eigenvalues define the hand synergies. A detailed description of this process is reported in [15]. Fig. 1, reproduced from [15], illustrates the coupling between some finger joints and the accumulated variance illustrating that the use of few PMDs is enough to describe a wide range of human hand postures. Fig. 2 shows some snapshots of the motion along the first PMD.

III. Proposed Optimization Cost Function

Optimization-based planners, like RRT*, use an optimization cost function to evaluate and compare different paths. In RRTs, the initial and the goal configurations are connected by piece-wise paths composed of a set of edges, called motions. When the only constraints are geometric, like in the case of mechanical hands, the paths can be piece-wise linear and the motions straight-line segments. In its simplest version the cost of a motion is equal to its length (which has to be minimized) or to the clearance (which has to be maximized), although other alternatives have also been proposed. Here the optimization function to evaluate the cost $C_{\boldsymbol{e}}$ of a motion $\boldsymbol{e}$ connecting two configurations is defined as a weighted sum of three components, i.e.:

Fig. 2. Configurations of the SAH hand when it is moved along the first PMD (i.e. with only one DOF). Reproduced from [21].

$$C_{\boldsymbol{e}} = w_d c_d + w_z c_z + w_h c_h, \tag{1}$$

where $w_d, w_h, w_z \in \mathbb{R}^+$ are weighting coefficients (that are currently fixed in an empirical way) and c_d, c_h, and c_z are the three components that evaluate, respectively:

- *The traveled distance*: c_d measures the length of the edge $\boldsymbol{e}$, with the aim of obtaining paths as short as possible. It is computed as the Euclidean distance in the configuration space $\mathcal{C}$:

$$c_d = |\boldsymbol{e}| \tag{2}$$

- *The zigzag*: c_z measures how much a motion $\boldsymbol{e}$ is aligned with the previous one, called the parent motion $\boldsymbol{e}_p$, with the aim of minimizing the zigzag of the path. c_z evaluates the alignment using the angle between the two consecutive edges $\boldsymbol{e}_p$ and $\boldsymbol{e}$, represented as $\widehat{\boldsymbol{e}\boldsymbol{e}_p}$, and multiplying it by the advance step ϵ used in the growing of the tree in order not to have too dissimilar magnitudes of the components of the cost $C_{\boldsymbol{e}}$:

$$c_z = \epsilon \;\; \widehat{\boldsymbol{e}\boldsymbol{e}_p} \tag{3}$$

  For motions along edges starting at the initial configuration, c_z is set to zero since in this case there is no parent edge.
- *The human-likeness*: c_h measures how much the edge $\boldsymbol{e}$ is aligned with the directions in $\mathcal{C}$ that describe the synergies, with the aim of obtaining paths with human appearance. Let:
  - $\mathcal{C}$ be the d-dimensional configuration space of the mechanical hand.
  - $\boldsymbol{u}_j \in \mathcal{C}$, with $j = 1, \ldots, d$, be the unitary vectors defining the PMDs (i.e. the eigenvectors resulting from the PCA) and $\lambda_1, \ldots, \lambda_d$ the corresponding eigenvalues ($\lambda_i > \lambda_{i+1} \forall i = 1, \ldots, d-1$).
  - $n < d$ be the number of PMDs to be used. n is selected such that the first n PMDs make the accumulated variance be above a given predefined

threshold. These PMDs represent the main couplings between the DOF of the finger joints.

– $\boldsymbol{e}_{\text{PMD}}$ be the weighted projection of $\boldsymbol{e}$ onto the subspace defined by the first n PMDs:

$$\boldsymbol{e}_{\text{PMD}} = \sum_{j=1}^{n} \frac{\lambda_j}{\lambda_1} (\boldsymbol{u}_j^T \cdot \boldsymbol{e}^T) \boldsymbol{u}_j, \tag{4}$$

Note that $|\boldsymbol{e}_{\text{PMD}}| \leq |\boldsymbol{e}|$ is always true.

Now, the cost c_h measures the alignment of $\boldsymbol{e}$ with respect to the vectors $\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n$ using the factor $|\boldsymbol{e}|/|\boldsymbol{e}_{\text{PMD}}|$. The minimum value of this factor is 1 and occurs when $\boldsymbol{e}$ is aligned with the first PMD. The more aligned with the main coupling directions an edge $\boldsymbol{e}$ is, the smaller its cost:

$$c_h = |\boldsymbol{e}| \min\left(\frac{|\boldsymbol{e}|}{|\boldsymbol{e}_{\text{PMD}}|}, C_c^{max}\right) \tag{5}$$

where C_c^{max} is a predefined saturation value that avoids useless large values of c_h when $|\boldsymbol{e}_{\text{PMD}}| \to 0$, and the multiplying factor $|\boldsymbol{e}|$ has been added to take into account the prolongation of this alignment/disalignment.

The objective is the minimization of the total cost C of a path defined in the RRT* by a sequence of motions $\boldsymbol{e}_i$:

$$C = \sum_{\boldsymbol{e}_i \in \text{Path}} C_{\boldsymbol{e}_i} \tag{6}$$

IV. Planning procedure

A. Sampling bias and node rejection

As mentioned in the introduction, the RRT* algorithm may have a slow cost improvement for problems with a high number of degrees of freedom. To mitigate this effect, sampling can be biased around the path in order to keep improving the first found solution while exploring the whole workspace to find the optimal one [9]. This proposal straightens and shortens the path by randomly choosing a node of the solution path and steering it a given amount towards the midpoint between the previous and the next nodes. A similar idea was proposed by the RRT*-Smart algorithm [22]. In this approach, once the first solution is found, a path optimization is run connecting the nodes of the path that are visible to each other, and the nodes of this optimized path are beacons where to oversample around. Also, in order to improve performance, the tree can be kept as reduced as possible by rejecting those samples that may be useless in finding a better solution, e.g. a sampled configuration $\boldsymbol{q}_{\text{rand}}$ is rejected if the distance from $\boldsymbol{q}_{\text{ini}}$ to $\boldsymbol{q}_{\text{rand}}$ plus the distance from $\boldsymbol{q}_{\text{rand}}$ to $\boldsymbol{q}_{\text{goal}}$ is greater than the cost of the current path [9]. All these proposals assume that the optimization parameter is the path length. We extend these ideas to consider any optimization function. Let consider the following functions:

- Nearest($V, \boldsymbol{q}$): Returns the closest node to configuration $\boldsymbol{q}$ from the set V of nodes of the tree.
- Steer($\boldsymbol{q}_1, \boldsymbol{q}_2, \epsilon$): Returns a new configuration $\boldsymbol{q}$ obtained by moving from $\boldsymbol{q}_1$ an small amount ϵ towards $\boldsymbol{q}_2$, or the final configuration $\boldsymbol{q}_2$ if the distance between $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ is smaller than ϵ.

Algorithm 1 Sample

Input: Configuration $\boldsymbol{q}_{\text{goal}}$
Path P
Probability α, β, γ
Radius r_{max}
Advance step ϵ
Node set V
Output: A sample to steer the RRT
if $P = NULL$ **then**
  if Rand$(0,1) < \alpha$ **then**
    $\boldsymbol{q}_{\text{rand}} \leftarrow \boldsymbol{q}_{\text{goal}}$
  end if
  $\boldsymbol{q}_{\text{rand}} \leftarrow$ SampleUniform($\mathcal{C}$)
else
  if Rand$(0,1) < \beta$ **then**
    n=RandPathNode(P)
    $\boldsymbol{q}_{\text{mid}} = (P(n-1) + P(n+1))/2$
    $\boldsymbol{q}_{\text{rand}} \leftarrow$ SampleUniform($\boldsymbol{q}_{\text{mid}}$, Rand$(0, r_{max})$)
  else
    if Rand$(0,1) < \gamma$ **then**
      while true **do**
        $\boldsymbol{q}_{\text{rand}} \leftarrow$ SampleUniform($\mathcal{C}$)
        $\boldsymbol{q}_{\text{near}} \leftarrow$ Nearest($V, \boldsymbol{q}_{\text{rand}}$)
        $\boldsymbol{q}_{\text{new}} \leftarrow$ Steer($\boldsymbol{q}_{\text{near}}, \boldsymbol{q}_{\text{rand}}, \epsilon$)
        if Cost($\boldsymbol{q}_{\text{near}}$) + EdgeCost($\boldsymbol{q}_{\text{near}}, \boldsymbol{q}_{\text{new}}$)+ EdgeCost($\boldsymbol{q}_{\text{new}}, \boldsymbol{q}_{\text{goal}}$) $<$ Cost($\boldsymbol{q}_{\text{goal}}$) **then**
          break
        end if
      end while
    else
      $\boldsymbol{q}_{\text{rand}} \leftarrow$ SampleUniform($\mathcal{C}$)
    end if
  end if
end if
return $\boldsymbol{q}_{\text{rand}}$

- EdgeCost($\boldsymbol{q}_1, \boldsymbol{q}_2$): Evaluates the cost $C_{\boldsymbol{e}}$ of the edge $\boldsymbol{e}$ that connects $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ using Eq. (1) .
- Cost($\boldsymbol{q}$): Returns the cost C from $\boldsymbol{q}_{\text{init}}$ to $\boldsymbol{q}$ using Eq. (6).
- Rand(v_1, v_2): A random value in the range $[v_1, v_2]$.
- SampleUniform($\mathcal{C}$): Returns a random configuration of $\mathcal{C}$ using a uniform sampling distribution.
- SampleUniform($\boldsymbol{q}, r$): Returns a random configuration inside the ball of radius r centered at $\boldsymbol{q}$ using a uniform sampling distribution.
- RandPathNode(P): A random node of the path P.

Algorithm 1 shows the sampling procedure, where: a) while a path is not found, it biases the search towards the goal with a probability α, as usually done in RRT-like algorithms; b) once the path is found, it biases the search around the path with a probability β to locally optimize the solution; c) configurations uniformly sampled from $\mathcal{C}$ are filtered, with a probability γ, to discard those that are not promising in finding a better solution, thus keeping the tree with a reduced size.

B. The planning algorithm

The proposed planning procedure is described in Algorithm 2, where the following functions are also used:

- k-Near($V, \boldsymbol{q}$): Returns the k nearest neighbors of $\boldsymbol{q}$ from

Algorithm 2 RRT* algorithm that uses the cost function described in Section III and the sampling procedure of Algorithm 1.

Input: Configurations $\boldsymbol{q}_{\text{init}}, \boldsymbol{q}_{\text{goal}}$
Probability P_{goal}, P_{path}, P_{filter}
Radius r_{max}
Advance step ϵ
Output: A path from $\boldsymbol{q}_{\text{init}}$ to $\boldsymbol{q}_{\text{goal}}$
$V \leftarrow \{\boldsymbol{q}_{\text{init}}\}; E \leftarrow \emptyset$;
for $i = 0$ **to** n **do**
$\boldsymbol{q}_{\text{rand}} \leftarrow$ Sample($\boldsymbol{q}_{goal}$, Path($\boldsymbol{q}_{\text{goal}}$), P_{goal}, P_{path}, P_{filter}, r_{max}, ϵ, V)
$\boldsymbol{q}_{\text{near}} \leftarrow$ Nearest($V, \boldsymbol{q}_{\text{rand}}$)
$\boldsymbol{q}_{\text{new}} \leftarrow$ Steer($\boldsymbol{q}_{\text{near}}, \boldsymbol{q}_{\text{rand}}, \epsilon$)
if CollisionFree($\boldsymbol{q}_{\text{near}}, \boldsymbol{q}_{\text{new}}$) **then**
$Q_{\text{near}} \leftarrow k$-Near($V, \boldsymbol{q}_{\text{new}}$)
$V \leftarrow V \cup \{\boldsymbol{q}_{\text{new}}\}$
$\boldsymbol{q}_{\text{min}} = \boldsymbol{q}_{\text{near}}$
$c_{\text{min}} =$ Cost($\boldsymbol{q}_{\text{near}}$) + EdgeCost($\boldsymbol{q}_{\text{nearest}}, \boldsymbol{q}_{\text{new}}$)
for all $\boldsymbol{q} \in Q_{\text{near}}$ **do**
if CollisionFree($\boldsymbol{q}, \boldsymbol{q}_{\text{new}}$) $\wedge$
Cost($\boldsymbol{q}$) + EdgeCost($\boldsymbol{q}, \boldsymbol{q}_{\text{new}}$) $< c_{\text{min}}$ **then**
$\boldsymbol{q}_{\text{min}} \leftarrow \boldsymbol{q}$
$c_{\text{min}} \leftarrow$ Cost($\boldsymbol{q}$) + EdgeCost($\boldsymbol{q}, \boldsymbol{q}_{\text{new}}$)
end if
end for
$E \leftarrow E \cup \{\boldsymbol{q}_{\text{min}}, \boldsymbol{q}_{\text{new}}\}$ //Connect along a minimum-cost path
for all $\boldsymbol{q} \in Q_{\text{near}}$ **do**
if collisionFree($\boldsymbol{q}_{\text{new}}, \boldsymbol{q}$) $\wedge$
Cost($\boldsymbol{q}_{new}$) + EdgeCost($\boldsymbol{q}_{\text{new}}, \boldsymbol{q}$) $<$ Cost($\boldsymbol{q}$) **then**
$E \leftarrow (E \backslash \{\text{Parent}(\boldsymbol{q}), \boldsymbol{q}\}) \cup \{\boldsymbol{q}_{\text{new}}, \boldsymbol{q}\}$ //Rewire
end if
end for
if $\boldsymbol{q}_{\text{new}} = \boldsymbol{q}_{\text{goal}}$ **then**
return Path($\boldsymbol{q}_{\text{goal}}$)
end if
end if
end for
return $\emptyset$

the set V of nodes of the tree (the value of k depends on the dimension of the configuration space and on the number of vertices of the tree, as detailed in [7]).

- Path($\boldsymbol{q}$): Returns a piece-wise rectilinear path, composed of edges of the tree, that connects the root node $\boldsymbol{q}_{\text{init}}$ to node $\boldsymbol{q}$. The path is computed by backtracking from $\boldsymbol{q}$ following the parent relationship in the tree.
- CollisionFree($\boldsymbol{q}_1, \boldsymbol{q}_2$): Returns true if the rectilinear edge in $\mathcal{C}$ connecting $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ is collision-free, and false otherwise.

It should be remarked that this RRT*: a) uses Eq. (1) and (6) for the cost function that allows to optimize distance, zigzag and human-likeness, and b) uses Algorithm 1 to obtain samples in an efficient way by biasing and filtering.

C. Implementation issues

The proposal has been implemented within The Kautham Project [21], a motion planning and simulation environment used at the Institute of Industrial and Control Engineering (IOC-UPC) for teaching and research. The core of the planners provided belong to the Open Motion Planning Library (OMPL) [23], which codes a wide set of the state-of-the-art sampling-based motion planning algorithms at an abstract level, i.e. without including issues related to robot modelling, collision-check or visualization. The main features of Kautham are the following: it allows to easily and flexibly use and parameterize many planners, it allows the visualization of 2D and 3D configuration spaces (and of 2D or 3D projections of the higher dimensional ones), it has an easy way to define coupled degrees of freedom (described below) and tailor the planning accordingly, it includes dynamic simulation, and it facilitates the integration with task planners and the benchmarking of planners. The cost function proposed has been coded as a class derived from the OMPL OptimizationObjective class and the planning algorithm as a derived class from the OMPL RRT* class in order to include the sampling bias and the filtering.

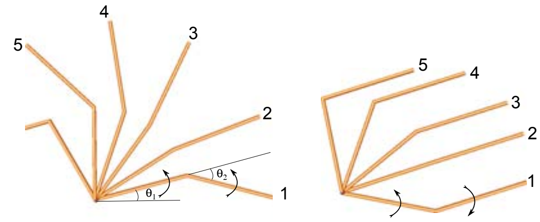

Fig. 3. Left: Motion of the 2 DOF planar robot along the first PMD that defines a positive coupling between the two joint angles θ_1 and θ_2, i.e. both joints move in the positive (counterclockwise) sense. Right: Motion along the second PMD that defines a negative coupling, i.e. θ_1 moves counterclockwise while θ_2 moves clockwise.

The coupling between the DOF of a robotic system is defined as follows [21]. Let $\hat{\boldsymbol{q}}$ be a d-dimensional configuration with the joint values set in the range $[0, 1]$. Then, the following expression is used to determine $\hat{\boldsymbol{q}}$ using a vector of p controls, $(c_1, \dots, c_p)^T$, a $d \times p$ mapping matrix K, and a vector of offsets, $(o_1, \dots, o_d)^T$:

$$\hat{\boldsymbol{q}} = \begin{bmatrix} \hat{q}_1 \\ \vdots \\ \hat{q}_d \end{bmatrix} = K \begin{bmatrix} c_1 - 0.5 \\ \vdots \\ c_p - 0.5 \end{bmatrix} + \begin{bmatrix} o_1 \\ \vdots \\ o_d \end{bmatrix} \tag{7}$$

Controls and offsets take values in the range $[0, 1]$, and the values $\hat{q}_i$ are forced to lie also in this range, i.e. whenever the i-th element of the right-hand side of Eq. (7) is greater than 1 then $\hat{q}_i = 1$ and whenever it is below 0 then $\hat{q}_i = 0$.

The mapping matrix K defines how the degrees of freedom are actuated, e.g. an identity mapping matrix indicates that all the degrees of freedom are independently actuated, a mapping matrix with some zero rows indicates that some degrees of freedom are not actuated (i.e. fixed), and a mapping matrix with columns with several non-zero elements indicates that some degrees of freedom are coupled, like in the case of PMDs.

V. Examples

A. A simple 2-DOF example

Let consider a 2-DOF planar robot with two rotational joints, and let define a coupling between them using the

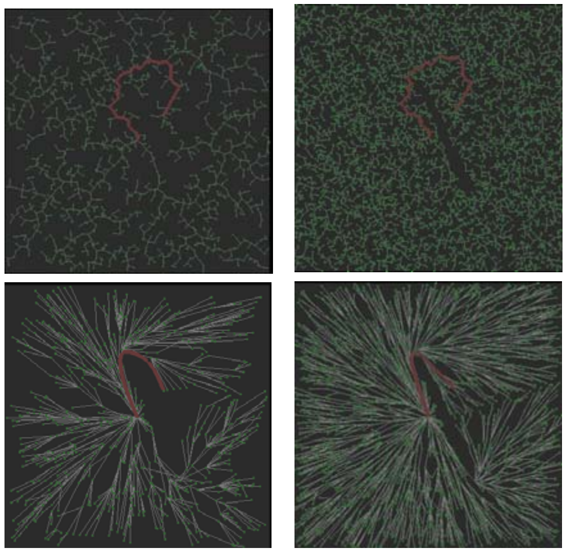

Fig. 4. RRT (top) and RRT* (bottom) for different number of samples.

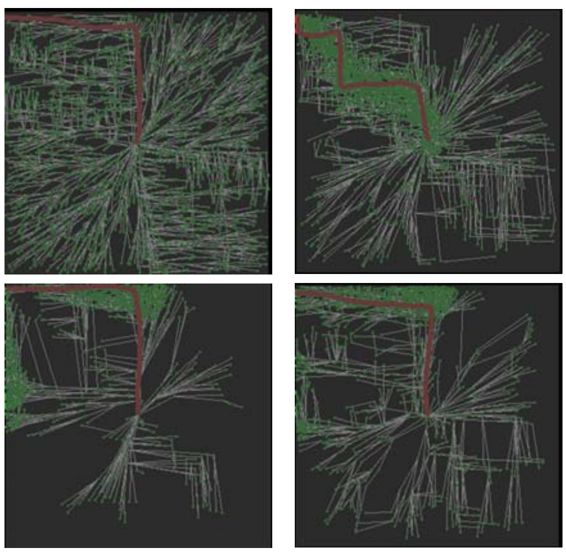

Fig. 5. RRT* optimizing the proposed cost: (top-left) plain solution; (top-right) solution using only local bias; (bottom-left) solution using only node filtering; (bottom-right) solution using local bias and node filtering.

following two PMDs:

$$\boldsymbol{u}_1 = \frac{1}{\sqrt{2}}[1, 1]^T \quad \lambda_1 = 1.0 \tag{8}$$

$$\boldsymbol{u}_2 = \frac{1}{\sqrt{2}}[1, -1]^T \quad \lambda_2 = 0.1 \tag{9}$$

i.e. the first PMD defines a positive coupling between joints, as shown in Fig. 3 (left) and the second one a negative coupling, as shown in Fig. 3 (right).

Fig. 4 shows the solutions found for a given query in an environment with a single obstacle using the standard RRT algorithm (top row) and the RRT* optimizing distance (bottom row) with an increasing number of samples. Observe that, as reported in [7], the RRT solution does not improve with an increasing number of samples while RRT* does.

For the case of an obstacle-free environment, Fig. 5 shows

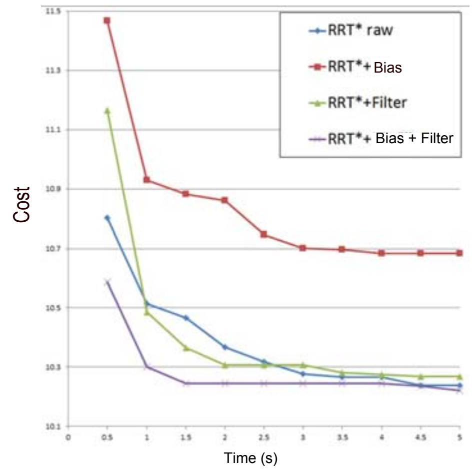

Fig. 6. Cost decrease as a function of time (in seconds) resulting form the average of 5 executions per planner.

a query that goes from the center of the configuration space towards the top-left corner, i.e. the rectilinear motion between the initial and the goal configurations is along the second PMD that defines the least desired coupling direction. Fig. 5 top-left shows the standard RRT* obtained using the proposed cost function with $w_d = 0.1$, $w_z = 1.0$ and $w_h = 1.0$. It can be seen that the solution path (in red) has a maximum projection along the first PMD. The other three snapshots illustrate the use of the same optimization function with: the path bias ($P_{\text{path}} = 0.8$ and $P_{\text{filter}} = 0.0$), the sample filtering ($P_{\text{path}} = 0.0$ and $P_{\text{filter}} = 1.0$) and the consideration of both bias and filtering ($P_{\text{path}} = 0.1$ and $P_{\text{filter}} = 0.8$). In all cases P_{goal} was set to 0.1. Fig. 6 shows how the cost keeps decreasing as time goes by and illustrates that the exclusive use of the path bias produces a local optimum with cost around 10.7, and that the combined use of path bias and sample filtering results in the fastest decrease of the cost.

The same problem as in Fig. 4 is illustrated in Fig. 7 with an RRT* optimizing distance (top) and optimizing the proposed cost function (bottom), and without using bias nor filtering (left figures) and with $P_{\text{path}} = 0.1$ and $P_{\text{filter}} = 0.8$ (right figures). It can be appreciated that in this latter case the tree has grown towards regions relevant to the query.

B. A 13-DOF mechanical hand

The proposed planning procedure was used to plan the motions of the Schunk SAH mechanical hand. This four-finger hand has four joints and three DOF per finger (the middle and distal phalanxes are mechanically coupled), and the thumb base has one extra DOF, therefore the hand has a total of 13 DOF. Fig. 8 shows snapshots of the motions of the SAH hand between two given configurations, the initial on the left and the goal on the right. Note that at these two configurations the positions of the index and the ring fingers are the same. The top row illustrates the solution found using an RRT* optimizing the distance. The motions of the shortest path found do not move at all the index nor the ring finger, thus resulting in a short but not human-like solution. On the other hand, the bottom row shows the solution found using

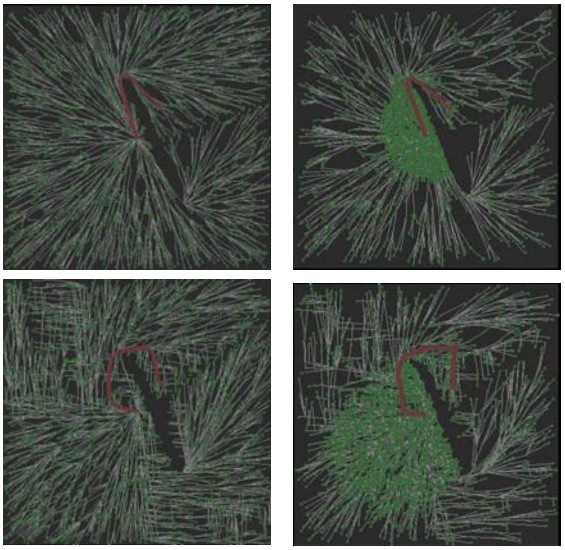

Fig. 7. RRT* optimizing distance (top): plain solution and solution using path bias and node filtering; RRT* optimizing the proposed cost (bottom): plain solution and solution using path bias and node filtering;

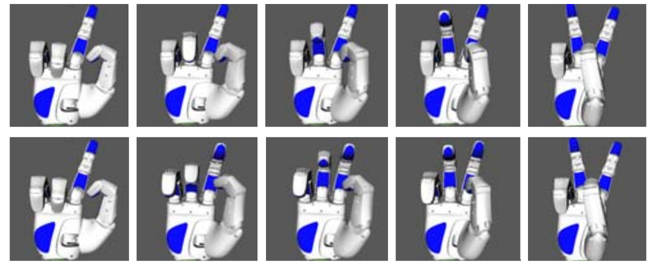

Fig. 8. Snapshots of a hand movement computed using an RRT* optimizing distance (top) and optimizing the proposed cost function (bottom).

an RRT* optimizing the proposed cost function. Human-like motions tend to couple positively the flexing of the index, middle and ring fingers (it is difficult for a human to flex one of them while keeping the others completely non-flexed). This is noted in the snapshots in the bottom row where it can be seen that: a) the index finger first flexes a little bit in order to couple positively with the middle finger along the last part of the motion; b) the ring finger is first extended with a move positively coupled with the middle finger and then flexes towards its final configuration. Therefore, it can be seen that the proposed cost function quantitatively evaluates the human-likeness of the mechanical hand motions and can effectively be used in optimization-based planners.

VI. CONCLUSIONS

This paper has presented a motion planner for robotic hands that tend to obtain human-like motions. The planner is based on the RRT* algorithm with an optimization function that measures and optimizes the alignment of the motions with the directions defined by the main synergies of the human hand. The proposal also includes some improvements for the RRT* in order to work in the high dimensional configuration space of the hand: a bias towards the path and a node filtering technique. Future work is directed towards the planning of human-like motions for a dual-arm robotic system.

REFERENCES

[1] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. K. Overmars. Probabilistic roadmaps for path planning in high - dimensional configuration spaces. *IEEE Trans. on Robotics and Automation*, 12(4):566–580, August 1996.

[2] J. J. Kuffner and S. M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 995–1001, 2000.

[3] R. Geraerts and M. H. Overmars. Sampling techniques for probabilistic roadmap planners. *Intelligent Autonomous Systems*, 8:600–609, 2004.

[4] D. Hsu, J.-C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. *The Int. J. of Robotics Research*, 25(7):627 – 643, 2006.

[5] D. Berenson, S. Srinivasa, D. Ferguson, and James Kuffner. Manipulation planning on constraint manifolds. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 625 – 632, 2009.

[6] M. Stilman. Global manipulation planning in robot joint space with task constraints. *IEEE Trans. on Robotics*, 26(3):576–584, 2010.

[7] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The Int. J. of Robotics Research*, 30(7):846–894, June 2011.

[8] A. Shkolnik and R. Tedrake. Path planning in 1000+ dimensions using a task-space voronoi bias. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 2892–2898, 2009.

[9] B. Akgun and M. Stilman. Sampling heuristics for optimal motion planning in high dimensions. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2640 – 2645, 2011.

[10] J. Röwekämper, G. D. Tipaldi, and W. Burgard. Learning to guide random tree planners in high dimensional spaces. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1752 – 1757, 2013.

[11] M. Santello, M. Flanders, and J. F. Soechting. Postural hand synergies for tool use. *J. of Neuroscience*, 18(23):10105–10115, December 1998.

[12] M. T. Ciocarlie and P. K. Allen. Hand posture subspaces for dexterous robotic grasping. *The Int. J. of Robotics Research*, 28(7):851–867, July 2009.

[13] D. Prattichizzo, M. Malvezzi, and A. Bicchi M. Gabiccini. On motion and force controllability of precision grasps with hands actuated by soft synergies. *IEEE Trans. on Robotics*, 29(6):1440–1456, 2013.

[14] L. Colasanto, R. Suaŕez, and J. Rosell. Hybrid mapping for the assistance of teleoperated grasping tasks. *IEEE Trans. on Systems, Man and Cybernetics: Systems*, 43(2):390–401, 2013.

[15] J. Rosell, R. Suárez, C. Rosales, and A. Pérez. Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures. *Autonomous Robots*, 31(1):87–102, 2011.

[16] S. Dalibard and J. Laumond. Linear dimensionality reduction in random motion planning. *The Int. J. Robotics Research*, 30(12):1461–1476, 2011.

[17] J. Rosell, R. Suárez, and A. Pérez. Path planning for grasping operations using an adaptive PCA-based sampling method. *Autonomous Robots*, 35(1):27–36, 2013.

[18] J. Rosell and R. Suárez. cRRT*: Planning loosely-coupled motions for multiple mobile robots. In *Proc. of the IEEE Int. Conf. on Emerging Technologies and Factory Automation, ETFA'14*, 2014.

[19] G. Gioioso, G. Salvietti, M. Malvezzi, and D. Prattichizo. Mapping synergies from human to robotic hands with dissimilar kinematics: An approach in the object domain. *IEEE Trans. on Robotics*, 29(4):825 – 837, 2013.

[20] Schunk GmbH & Co. KG. Schunk anthropomorphic hand. http://www.schunk.com/, May 2006.

[21] J. Rosell, A. Pérez, A. Aliakbar, Muhayyuddin, L. Palomo, and N. García. The Kautham Project: A teaching and research tool for robot motion planning. In *Proc. of the IEEE Int. Conf. on Emerging Technologies and Factory Automation, ETFA'14*, 2014.

[22] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan. RRT*-smart: Rapid convergence implementation of RRT*; towards optimal solution. In *Mechatronics and Automation (ICMA), 2012 Int. Conf. on*, pages 1651–1656, 2012.

[23] I. A. Şucan, M. Moll, and L. E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012.