Towards a Data-driven Approach to Motion Planning

Arjun Menon[1], Pooja Kacker[2], Sachin Chitta[1]

***Abstract*— Co-robots, i.e. robots that work close to people, will need to account for the preferences and expectations of their human co-workers in executing trajectories or actions. Consistent, legible and predictable trajectories are a key factor in making humans comfortable around robots. In this work, we take a data-driven approach towards designing robot trajectories that are more acceptable to human co-workers and observers. We use an online survey to ask people to rate multiple robot trajectories generated in a variety of environments. We compute a large set of features for each trajectory, also taking into account environment information. We use a combination of the features and the survey ratings to learn a classifier that predicts the rating for a new trajectory based on the learned human-observer preferences. The classifier also helps identify and highlight the most important features that influence people's ratings of the trajectories. Finally, we discuss how a data-driven approach using the results of this analysis can be used to help design better trajectories that are more acceptable to people.**

I. INTRODUCTION

Robotic motion generation in industrial environments does not take into account any notion of the acceptability of trajectories for human observers. The need for more flexibility in manufacturing and logistics, though, has resulted in a greater need for robots that can work beside humans. As robots and people start becoming co-workers, taking human perception of robotic paths into account will become more important. Humans are able to predict and adjust their trajectories to account for the expectations and trajectories of their co-workers easily. For human co-workers to be comfortable, robots will similarly have to tailor their trajectories for the presence and actions of the people around them.

In this work, we take a first step towards the ultimate goal of generating robotic arm trajectories in cluttered environments that are acceptable for human observers and co-workers. We take a data-driven approach to this problem, using an online survey where human observers rate robot motion quality to generate a large dataset of human-observer preferences for robot trajectories. We use a feature set computed for each trajectory to then learn a relationship between the features in the trajectory and the rating that it gets from the human observers. A classifier learned in this manner serves two purposes: (a) it helps in identifying a set of features that are the most important in predicting human preferences for robot trajectories and (b) it could form the basis of a new approach to generating trajectories that is driven by learned human-observer preferences.

*This work was partly supported by NSF NRI Grant IIS 1317849

[1]SRI International, Menlo Park, CA arjun.menon@sri.com, sachin.chitta@sri.com

[2]Independent Consultant, Menlo Park, CA

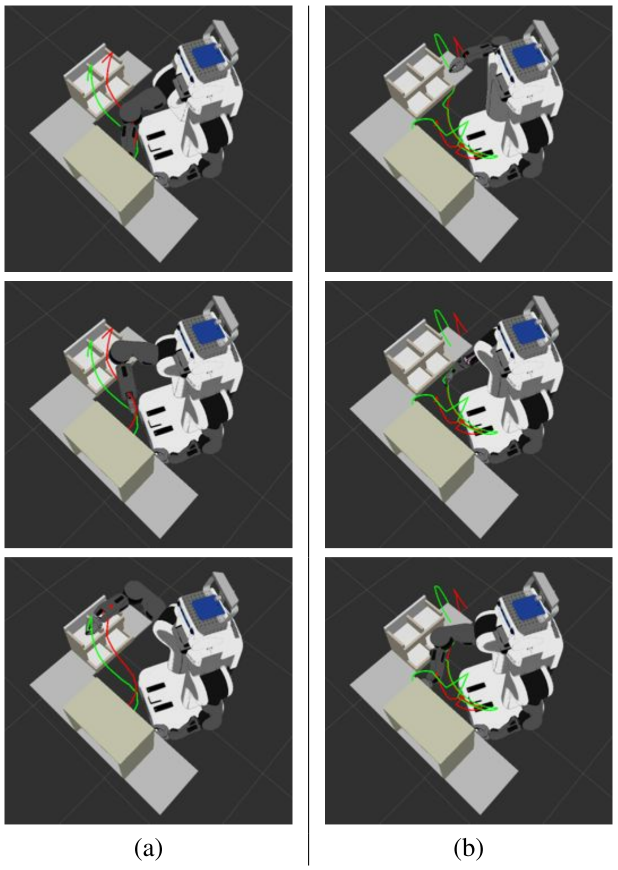

Fig. 1: Example trajectory snapshots for motions that survey participants rated (a) highly and (b) poorly. In (b) the hand of the robot ducks under the table in between the first and middle snapshot.

This paper is organized as follows. In Section II, we first describe related work in this area. In Section III, we present details of the online survey including information about the types of trajectories and environments used. Section IV presents the results from the survey and analysis, including the trajectory features used for classification and Section V covers the classification approaches used with the data. In Section VI, we conclude with a discussion of how our results could be used to design motion planners that generate more human-acceptable trajectories.

II. BACKGROUND

Robot motion has been primarily influenced by the need to generate smooth jerk-free trajectories for industrial robots that minimize wear and tear of internal parts. In recent years, there has been more interest in the generation of motions for robots working in human environments, i.e. environments with clutter and people working next to the robots. Most attention, though, has been focused on generating *collision-free* motions. Examples include the use of randomized planners [1], trajectory optimization techniques [2], and deterministic search-based techniques [3].

Human motions have been the subject of recent study with the intent of determining whether there are certain motor or locomotor invariants that best describe human manipulation or locomotion. Human motion has been observed to be stereotypical [4], i.e. learned motion patterns that are repeatedly used. This requires less planning of individual motions and contributes to making human motions more deterministic and predictable. The minimum jerk principle for human motions specifies that the trajectory of the human can best be represented as a motion that minimizes jerk of the hand [5]. Uno et. al. [6] used minimum torque change as a measure to explain the observed human motions. Minimizing jerk at the joint level [7] has been another criterion used for human motion. Human motion has also been found to be optimized [8]. Albrech et. al. [9] studied the reaching motions of several subjects in table-setting situations and attempted to codify the motion using cost functions representing minimum-jerk hand motions, minimum-jerk joint motions, minimum torque change and a combination of all three. They found that no single combination of the cost functions explained the observed data completely.

Recent work has also addressed the task of making motions predictable (similar motions in similar environments and tasks) and/or legible (make it easier for human observers to understand the intent of the robot) [10]. Legibility makes it possible for human observers to understand clearly what a robot is intending to do, allowing them to modify their actions and motions if necessary. Predictability implies that the robot will always be consistent in its actions, removing the element of surprise for human observers. These concepts have been well-studies recently with the intent of measuring legibility [11], using expressions to improve the readability of robots [12] and generating anticipation [13]. In [14], the notion of legibility was incorporated into a constrained trajectory optimization motion planner to generate motions that are more legible to human observers.

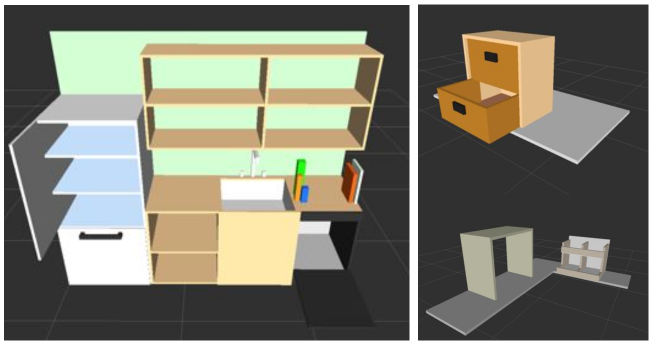

Fig. 2: The scenes used in generating the robot motions to collect feedback from human observers. *Drawer* (top right), *Countertop* (left), and *Industrial* (bottom right)

There has been relatively little examination of how human observers perceive robot arm trajectories in environments with obstacles. There is no data available for how human observers react to such robot motions. It is not clear whether minimum-jerk trajectories (or other cost functions) are the best descriptors of motions in such environments or if there are other features that better explain human-observer preferences in such sitations. Our goal in this work is to collect the data needed to answer the query, “Which robot trajectories in cluttered environments do human-observers prefer” A second question that we would like to answer is “Why are certain robot trajectories preferred by human-observers?” This is a larger question that we aim to address in future work and instead we focus on, “Can we predict a human-observer preference for a given trajectory” The answer to this latter question, we hope, will eventually point us towards trajectories that are better accepted by human observers.

III. Approach

Our approach starts with an online survey designed to get information from multiple users about how well they like robot trajectories. In this section, we will describe in detail the infrastructure used to create the trajectories and movies that were used in the survey. We will also describe the online survey procedure in detail.

A. Planning System Infrastructure

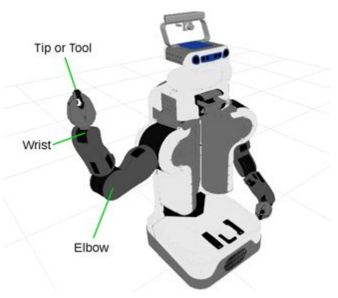

Fig. 3: PR2 robot.

We extended the MoveIt! benchmarking architecture [15] with video generation capabilities to automate the generation of a large number of trajectories. The MoveIt! architecture allows for the definition of environments, referred to as planning scenes, start and goal states for robots (motion planning queries). It also allows for the use of any supported planning algorithms (planners). A combination of different scenes, motion queries, and planners was used to generate robot trajectories for evalutation. We chose three distinct scenes (Fig. 2): a kitchen (Countertop), an industrial workbench (Industrial) and a drawer on a table (Drawer). Multiple queries were generated in each scene for trajectories of the right arm of the PR2 robot (Fig. 3. The robot was positioned in the scenes manually and motion planning queries were constructed by hand. The positioning in the scenes was done with the expectation that the planning queries would not be impossible or trivial for the planning algorithms to solve.

The planning algorithms that we used are RRT-Connect [1] and RRT*. RRT* [16] was used with two different cost functions: PathLength and MaxMinClearance.The objective

Efficiency	Highly inefficient	Moderately inefficient	Slightly inefficient	Neutral	Slightly efficient	Moderately efficient	Highly efficient
Elegance	Highly awkward	Moderately awkward	Slightly awkward	Neutral	Slightly elegant	Moderately elegant	Highly elegant
Smoothness	Highly rough	Moderately rough	Slightly rough	Neutral	Slightly smooth	Moderately smooth	Highly smooth
Overall	Extremely poor	Moderately poor	Somewhat poor	Neutral	Somewhat good	Moderately good	Extremely good

TABLE I: The four rating categories/attributes and the associated scales for each presented to the survey respondent.

for PathLength minimizes the length of the trajectory (as the sum of the euclidean distances between ends of trajectory segments). The objective for MaxMinClearance computes uses the minimum clearance of a configuration belonging to a trajectory as the value it is trying to maximize. Both planners are implemented in the OMPL library [17]. All the paths generated by RRT-Connect are shortcutted for a finite amount of time to improve the quality of the paths. The planned trajectories are post-processed by an iterative-time parameterization method that time-parameterizes the trajectories while taking into account velocity and accelerations. The infrastructure we designed around MoveIt! allows us to automatically generate movies of the generated trajectories using the ROS Visualizer (Rviz). We generated movies from different viewpoints to maximize the visibility of the motion to human observers reviewing it. A total of 103 different trajectories were generated in this manner.

Our main focus in generating trajectories was to generate a wide sample. The choice of planners used to generate the trajectory was not important. We preferred the randomized planners since they natively generate a wide variety of plans that are less likely to be similar to each other. We could have chosen to generate all our trajectories with an asymptotically optimal planner like RRT* but we would not have been able to generate a variety of plans. This would also have restricted the richness of the trajectory features we could work with. Our intent in this work is not to evaluate particular planning algorithms but instead to examine the preference that human-observers have for generated trajectories regardless of the methods used to generate them.

B. Online Survey

We conducted an online survey[1] where we asked human observers to rate a variety of trajectories of the PR2 robot. The survey was built on Amazon's Mechanical Turk, which has become a useful tool for large online data gathering and crowd-sourced work. Each respondent in the survey was presented with a set of demographic questions, a video, and asked to rate the robot's motion on the attributes of elegance, efficency, smoothness and overall impression (a summary of the rating scales used in the main survey question can be seen in Table I). The ratings were later converted into numerical values on a scale of 1 to 7. Each respondent was expected to rate multiple trajectories but could not rate a trajectory more than once. Each video has four different views of the robot displayed simultaneously. Respondents were asked demographic questions about familiarity with video games, familiarity with robotics, age group and highest education level. The demographic information was not used to filter the responses we collected.

Mechanical Turk allows monetary compensation on completion of the survey, which we set to 15¢ per survey. We estimated the time that each respondent would take to rate a single trajectory as 1-2 minutes. Additionally, the respondents were allowed to replay the trajectory video repeatedly. The maximum possible payout for each respondent is thus 15.45$ (if each survey took 1.5 minutes to complete then the wage is approximately 6$ per hour). This is standard compensation for quality work on Mechanical Turk for tasks of this type, as was pointed out to us by several veteran Mechanical Turk surveyors.

Respondents on Mechanical Turk are only paid if the survey organizer approves their response. A control question, asking the respondent to identify the known action of a robot in a video, served as the first satisficing trap to ensure that respondents were truthfully attempting the survey (see [18] for more information on satisficing in survey responses). Survey responses were also manually examined to assess whether respondents were responding honestly. We looked at whether respondent responses were outliers when compared to the rest of the group. We also examined the average time taken by each respondent to complete the survey. Overall, we found that all the respondents had made a honest attempt at filling out the rating and we did not reject any responses.

IV. Survey Results and Analysis

The survey resulted in at least 20 ratings for each trajectory (so there were a total of 2060 ratings from participants in the survey). In this section, we will first analyze the raw results from the survey (which are interesting in themselves). Subsequently, we will present a set of features that were used to represent individual trajectories. These features will be used in the next section to learn the human-observer ratings for robot trajectories.

A. Raw Survey Results

The raw results of the survey are summarized in Fig. 4 which shows a histogram of averaged responses for each of the four survey questions across the 103 trajectories that were rated by users. It is clear that a majority of the trajectories in our survey were well received by the human observers but there were quite a few trajectories that were rated badly as well. An analysis of the ratings for each trajectory showed that four (averaged) ratings for each individual trajectory were always similar to each other, indicating that users had similar preferences for each trajectory regardless of the individual attributes used to describe the preference for the trajectory (efficient, elegant, smooth, overall). Table II shows the Pearson's correlation coefficients corresponding to pairs

[1]The online survey was conducted after gaining a waiver from SRI's IRB after a review of our survey procedure. Our survey followed standard guidelines for privacy of respondent information when conducting a survey on Mechanical Turk.

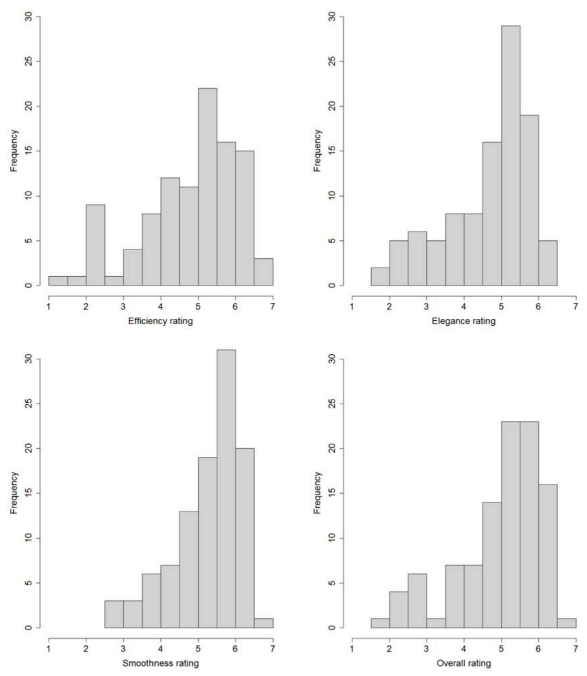

Fig. 4: Histogram of averaged ratings for the 4 different attributes for 103 trajectories. Clockwise from top left: Efficient vs. Inefficient, Elegant vs. Awkward, Smooth vs. Rough, Overall Rating

	Efficient	Elegant	Smooth	Overall
Efficient	1	0.96	0.91	0.98
Elegant	0.96	1	0.96	0.98
Smooth	0.91	0.96	1	0.96
Overall	0.98	0.98	0.96	1

TABLE II: Correlation between the four attributes in the survey.

of attributes. Attributes across individual questions are highly correlated at a statistically significant level of 1% (with p-values $< 0.01\%$). The Spearman rank correlation coefficients are also high and statistically significant.

B. *Trajectory Features*

We identified and computed multiple types of trajectory features. The first set of features we compute for each trajectory take into account the environment around the robot, the shape of the trajectory and the dynamics of the trajectory. These make up our unary features computed only on the input trajectory. We also include features that compare each trajectory to a nominal trajectory - these feature serve to characterize how much the trajectory deviates from the "expected" path. We go over these comparative features in the next section. First, we detail the features we computed on each trajectory ξ with waypoints q_i where $i = 0, \dots, n$ and $n = |\xi|$. Note that our features are generic, i.e. they can be computed in the same manner for any robot arm trajectory. We believe that our methods should be easily extensible to robots other than the PR2 as well and individual trajectory classifiers could be built for each different robot arm in a similar manner.

Path length: We compute path length for the joint-space trajectory and for three workspace trajectories for the elbow (ξ_{elbow}), wrist (ξ_{wrist}) and finger-tip (ξ_{tip}) links of the right arm (the moving arm). We also compute the length of the trajectory in angular distance by summing over the waypoints $\arccos(\langle Q_{\text{i}}, Q_{\text{i+1}}\rangle^2 - 1)$ where Q_{i} is the quaternion orientation for the i-th waypoint (referred to later as the "quat_dist" feature).

Clearance: The clearance of a trajectory ξ is the average over the waypoints of the minimum distance of configurations to the closest obstacle.

Smoothness: The smoothness of a trajectory is computed by averaging the included angle of three consecutive waypoint configurations, for all triplets of waypoints in ξ. The angle is computed using joint-space euclidean distance between configurations as the length of sides of the triangle as described in [15]. We compute smoothness for both the joint-space trajectory ξ and for workspace trajectories ξ_{elbow}, ξ_{wrist}, and ξ_{tip}.

Maximum Acceleration: We compute the absolute maximum acceleration for the joint-space trajectory ξ and for workspace trajectories ξ_{elbow}, ξ_{wrist}, and ξ_{tip}.

Maximum Radius of Curvature: Curvature at a given waypoint is computed by splines to the workspace trajectory. We compute maximum curvature for only the workspace trajectories ξ_{elbow}, ξ_{wrist}, and ξ_{tip}.

Joint Limit Distance: The maximin distance to joint limits is the maximum over waypoints, of the miniminum over joints of their distance to their respective closest limit (upper or lower), or $\max_{q\in\xi} \min_{\theta\in q} \min(|\theta - \theta_U|, |\theta - \theta_L|)$.

C. *Comparative Trajectory Features*

We also wanted to design features that captured the difference between the planned trajectory and an expected path for the end-effector in the environment. The expected or nominal end-effector trajectories were created using a breadth-first search in a 3D voxel approximation of the planning problem. The environment model was first discretized into a voxel-grid, and voxels are marked as occupied or empty. Any voxel that is partially occupied is considered to be occupied. A 26-connected grid was used in the planning process to find a path in the voxel grid for a cubical shape moving from the start position of the end-effector of the arm to the desired goal position. Essentially, this would be the optimal path followed by the end-effector from start to goal if it were modeled as a free-standing cube. Next, we compute comparison features between ξ_{BFS} and three workspace trajectories for the elbow (ξ_{elbow}), wrist (ξ_{wrist}) and finger-tip (ξ_{tip}) links. The comparison features computed between each pair of trajectories are the Hausdorff distances [19] and Dynamic Time Warping [20] distances.

Feature Name	Metric	Feature Description
path_length, smoothness, clearance, max_lin_acc, joint_limit_distance	Joint-space	Features discussed in Section IV-B computed on joint-space trajectory ξ
[link]_length, [link]_smoothness, [link]_max_curvature, [link]_lin_acc, [link]_quat_dist	Cartesian	Features discussed in Section IV-B computed on 3D cartesian path traced by [link] in ξ
h_[link]_bfs, h_[link1]_[link2], H_[link]_bfs, H_[link1]_[link2]	Cartesian	Assymetric (h) or symmetric (H) hausdorff distance between 3D path of [link] w.r.t BFS path or between 3D paths of [link1] and [link2]
[link]_bfs_dtw, [link1]_[link2]_dtw	Cartesian	DTW cost between path of [link] and the BFS path, or between paths of [link1] and [link2]

TABLE III: The complete list of computed trajectory features used for classification.

Hausdorff Distance: The Hausdorff distance H_{AB} between trajectory ξ_{A} and ξ_{B} can be computed as follows:

$$\begin{aligned} H_{\text{AB}} &= \max(h_{\text{AB}}, h_{\text{BA}}) \\ h_{\text{AB}} &= \max_{q_{\text{A}} \in \xi_{\text{A}}} \min_{q_{\text{B}} \in \xi_{\text{B}}} \text{d}(q_{\text{A}}, q_{\text{B}}) \\ h_{\text{BA}} &= \max_{q_{\text{B}} \in \xi_{\text{B}}} \min_{q_{\text{A}} \in \xi_{\text{A}}} \text{d}(q_{\text{B}}, q_{\text{A}}) \end{aligned} \tag{1}$$

where q_{A} is a configuration (or in this case an end effector position) at the waypoint in trajectory ξ_{A} and $\text{d}(q_{\text{i}}, q_{\text{j}})$ is the euclidean distance between the trajectory waypoints.

We also use the Interpolation-based Modified Hausdorff distance [19], which is defined as:

$$\text{imh}_{\text{AB}} = \frac{1}{n_A} \sum_{q_{\text{A}} \in \xi_{\text{A}}} \min_{\overline{q_{\text{B,i}}} \in \xi_{\text{B}}} \text{d}(q_{\text{A}}, \overline{q_{\text{B,i}}}) \tag{2}$$

where n is the number of waypoints, $\overline{q_{\text{B,i}}}$ is the i-th segment in ξ_{B}, and $\text{d}(q, \overline{q})$ is the shortest distance between the point and the line segment. We calculate this distance only one way, for $(\xi_{\text{elbow}}, \xi_{\text{BFS}})$, $(\xi_{\text{wrist}}, \xi_{\text{BFS}})$. and $(\xi_{\text{tip}}, \xi_{\text{BFS}})$.

Dynamic Time Warping: Dynamic Time Warping (DTW) was previously used in a wide varity of applications ranging from speech recognition, gesture recognition, handwriting matching, and can be applied here to compare two different trajectories [20]. It provides a way to compute a distance despite differences in timing of the trajectories as well. DTW computes a mapping of a subset of the waypoints in ξ_{A} to a subset of the waypoints in ξ_{B} which constitutes the warping path, or

$$\xi_{\text{DTW}} = \{(0,0), (a_i, b_j), \ldots, (n_A, n_B)\} \tag{3}$$

where $n_A = |\xi_{\text{A}}|$ and $n_B = |\xi_{\text{B}}|$ which are the number of waypoints in the trajectory. More on the construction of the ξ_{DTW} can be read in [20]. The DTW feature is the cost of the warping path.

V. Classification

The survey data provides a great platform to answer the question, "Can we predict a human-observer preference for a given trajectory?". Our approach to this question is to let the data drive us towards the answer instead of trying to hand-design a cost function or set of features that offer the best explanation. Towards this end, we chose to learn a classifier based on the available data to help predict the human-observer rating for a new trajectory. The 103 data points (corresponding to the 103 distinct trajectories) were broken up into 3 classes based on average overall ratings:

- low - 11 data points with ratings $\in (1, 3]$,
- medium - 29 data points with ratings $\in (3, 5]$,
- high 63 data points with rating $\in (5, 7]$.

The data was divided into two samples, a training sample and a test sample, with 1/3rd of each class (overall rating) used to construct the test data sample and the remaining 2/3rd data points within each class used to train the various models. This results in a training set consisting of 68 data points and test set consisting of 35 data points. Table IV shows how the average overall ratings are distributed across training and test sets.

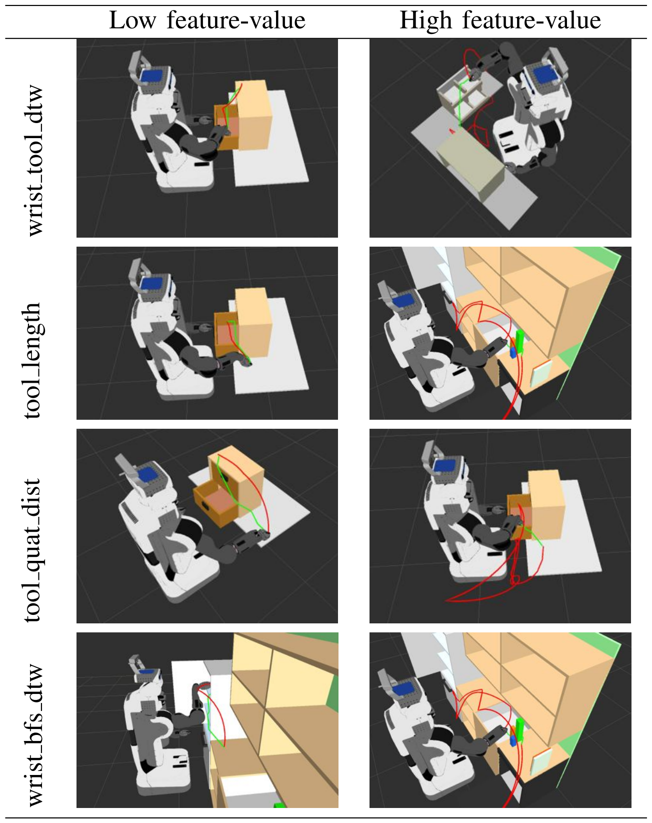

Fig. 5: Example trajectories for different values of the most useful features. Trajectories on the left have low feature values while trajectories on the right have high features values for each corresponding feature in that row.

	low (1,3]	medium (3,5]	high (5,7]
Training	7	19	42
Test	4	10	21

TABLE IV: Distribution between training and test samples.

In all, we utilized three different classification frameworks to predict overall ratings - decision trees, random forests and boosting. As a base framework we first construct a decision tree model. Decision trees for classification provide a good visual representation especially in the presence of complex features, where the relationships between predictors and response is not immediately obvious. The features of interest in the first decision tree that we derived include the DTW distance computed for the wrist (vs. the tool path) denoted by wrist_tool_dtw, the Hausdorf distance computed for the end-effector and the max curvature computed for the elbow. Using 10-fold cross-validation, we also design a simpler tree that has just three terminal nodes with the wrist_tool_dtw feature serving as the key discriminant among the classes. This tree has an accuracy of 86% on the test data. Large values of the wrist_tool_dtw feature correspond to extra rotations of the wrist as the tool is moving towards the goal - such trajectories are rated as low quality by the human observers.

The second classification framework we use is a random forest ensemble of trees. This involves randomly dividing the data into several bootstrap training samples and building a tree on each bootstrap sample. Within each boostrapped sample, at each split a subset of features is randomly selected from the full set of features, but only one predictor is used at each split. This brings additional randomness within each tree. The class is chosen by majority vote across all trees. For every tree constructed from a bootstrapped training sample there is one Out of Bag (OOB) error which can be thought of as a test error for every tree. It is estimated internally, during the run. On test data, the random forest method returns 83% accuracy.

Fig. 6 and Fig. 7 show two measures of variable (or feature) importance. The first measure shows the mean decrease in accuracy for each feature. The importance of a feature is determined by how much the accuracy of the random forest is affected by comparing, for each tree, the OOB error with the error obtained after permuting each feature. The differences are averaged across trees and normalized. The second measure is based on the Gini index and measures the total decrease in node heterogeneity/impurities from splitting on the feature.

For both measures higher values denote that the features are more important. Again, the wrist_tool_dtw feature stands out as one of the more important features in building the classifer. Two path lengths tool_length (the distance traveled by the tool) and path_length (the total distance traveled in joint space) also serve as important features - in most environments the total distance traveled by the arm between start and goal locations is expected to be small. The tool_quat_dist and wrist_quat_dist measure the rotation of the end-effector and it seems natural that excessive rotations of the end-effector would contribute to trajectories getting a bad rating. It is also clear that the BFS path, which serves as the nominal path for computing both the DTW and Hausdorf distance, is a good approximation of the path expected by the human observer for the end-effector of the robot.

The third classification framework that we use is Boosting. In the Boosting framework, the trees are grown sequentially,

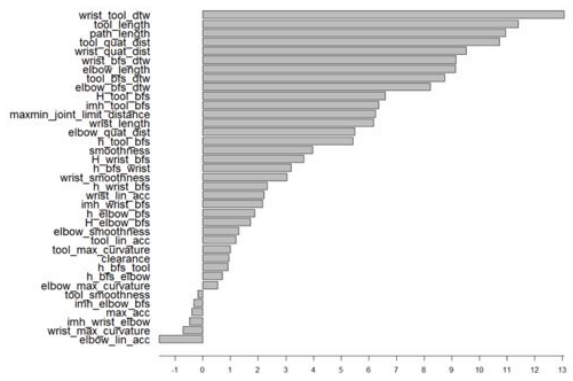

Fig. 6: Random Forest: Mean decrease in accuracy for each feature.

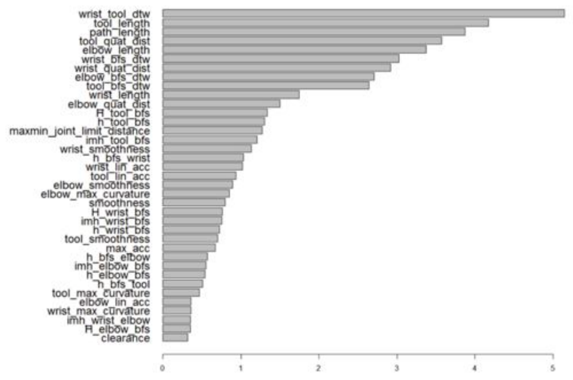

Fig. 7: Random Forest: Gini index (higher values indicate more important features).

using information (weights) from previously grown trees. The algorithm adaptively changes the distribution of the training data by iteratively reweighting each observation in each iteration. The weights are updated using (Breimans) weight updating coefficient which assigns higher weights to the wrongly classified observations and lower weights to the correctly classified examples. In every iteration, the classifier thus focuses on the most difficult examples. Using 10-fold cross validation with all the available data, we obtain a classification accuracy of 76.7%. Fig. 8 shows the relative importance of different features using the Boosting classifier. 7 of the top 10 features on this list are also part of the top 10 features using random forests.

The classification results clearly show the value in using a data-driven approach to determining human preferences for robot trajectories. It is difficult, in obstacle filled cluttered environments, to hand-design an appropriate cost function for generating human preferred trajectories. Tuning coefficients and weights for all the individual cost components for designing preferred trajectories requires significant effort. The classifiers that we have built give us a notion of the quality of trajectories being generated without an explicit need to model the cost function. The output from the classifier could be integrated into gradient-free motion planning schemes (e.g. STOMP or a simulated annealing approach) as a cost for

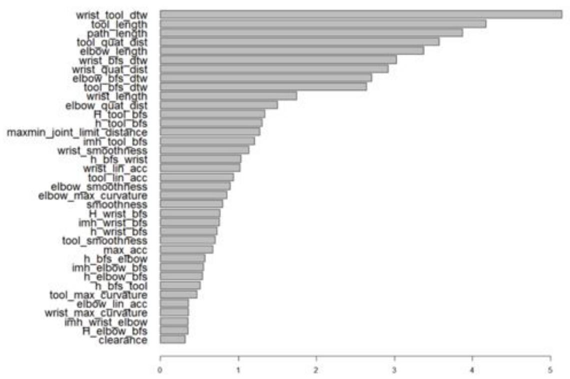

Fig. 8: Boosting: Gini index (higher values indicate more important features).

a particular trajectory. The identification of the key features that are important in defining a preferred trajectory should also help in the design of cost functions, especially since some the features may not be traditionally included as part of a cost function. In particular, note that a feature like wrist_tool_dtw is not used in typical optimization schemes like RRT* which rely on a path length criterion.

VI. Conclusion

In this work, we have gathered ratings for a large dataset of robot trajectories. We have already gained significant insights into the types of trajectories that are preferred by human observers. In particular, we have been able to identify a set of features that seem to influence the human's perception of the trajectories the most. This can help us in defining better cost functions when generating the trajectories. The classifiers that we have learned could also be used to directly inform gradient free motion planning algorithms by assigning a rating to new trajectories. Our results were generated using the PR2 but we believe that they will easily translate to other robot models as well. To this end, we will keep the survey open and attempt to collect data for a larger set of trajectories in more environments and with different robots.

The dataset we have collected and the analysis we have already done in this paper will serve as a platform for further development of the idea of using a data-driven approach to motion planning. We intend to attempt to learn a cost function for describing the relation between the features and the ratings of the generated trajectories using inverse reinforcement learning. We hope to examine whether a cost function derived in this manner will lead to generation of more preferred trajectories. We would also like to extend our set of features to include more dynamic features to capture the effect that the dynamics of the motions have on the perception of the motion. The infrastructure that we have developed for generating and conducting surveys is fairly generic and could easily be extended to surveying people using trajectories from real robots. This is another avenue we intend to explore in the near future. We also intend to explore the potential for open-sourcing the data and infrastructure to allow other researchers to conduct surveys of their own.

Acknowledgment

We would like to thank Leila Takayama for valuable advice on structuring the online survey. We would like to thank Lydia Kavraki and Mark Moll for valuable discussions and suggestions.

References

[1] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.

[2] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *International Conference on Robotics and Automation*, Shanghai, China, May 2011.

[3] B. J. Cohen, S. Chitta, and M. Likhachev, "Search-based planning for manipulation with motion primitives," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2902–2908.

[4] C. Atkeson and J. Hollerbach, "Kinematic features of unrestrained vertical arm movements," *Journal of Neuroscience*, vol. 5(9), pp. 2318–2330, 1985.

[5] T. Flash and N. Hogan, "The coordination of arm movements: An experimentally confirmed mathematical model," *Journal of Neurosicence*, vol. 5(7), 1985.

[6] Y. Uno, M. Kawato, and R. Suzuki, "Formation and control of optimal trajectory in human multi-joint arm movement," *Biological Cybernetics*, vol. 61(2), pp. 89–101, 1989.

[7] D. A. Rosenbaum, L. D. Loukopoulos, R. G. J. Meulenbroek, V. J., and E. S. E, "Planning reaches by evaluating stored postures," *Psychological Review*, vol. 102(1), pp. 28–67, 1995.

[8] G. Arechavaleta, J.-P. Laumond, H. Hicheur, and A. Berthoz, "Optimizing principles underlying the shape of trajectories in goal oriented locomotion for humans," in *International Conference on Humanoid Robots*, 2006.

[9] S. Albrecht, K. Ramirez-Amaro, F. Ruiz-Ugalde, D. Weikersdorfer, M. Leibold, M. Ulbrich, and M. Beetz, "Imitating human reaching motions using physically inspired optimization principles," in *11th IEEE-RAS International Conference on Humanoid Robots*, Bled, Slovenia, October, 26–28 2011.

[10] M. Beetz, F. Stulp, P. Esden-Tempski, A. Fedrizzi, U. Klank, I. Kresse, A. Maldonado, and F. Ruiz, "Generality and legibility in mobile manipulation," *Autonomous Robots*, vol. 28, pp. 21–44, 2010.

[11] C. Lichtenthäler, T. Lorenz, and A. Kirsch, "Towards a legibility metric: How to measure the perceived value of a robot," *ICSR Work-In-Progress-Track*, 2011.

[12] L. Takayama, D. Dooley, and W. Ju, "Expressing thought: Improving robot readability with animation principles," in *HRI*, 2011.

[13] M. Gielniak and A. Thomaz, "Generating anticipation in robot motion," in *RO-MAN*, 2011.

[14] A. Dragan, K. C. T. Lee, and S. Srinivasa, "Legibility and predictability of robot motion," in *ACM/IEEE International Conference on Human-Robot Interaction*, 2013.

[15] B. Cohen, I. A. Sucan, and S. Chitta, "A generic infrastructure for benchmarking motion planners," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 589–595.

[16] E. Frazzoli and S. Karaman, "Incremental sampling-based algorithms for optimal motion planning," *Int. Journal of Robotics Research*, 2010.

[17] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, http://ompl.kavrakilab.org.

[18] J. A. Krosnick, S. Narayan, and W. R. Smith, "Satisficing in surveys: Initial evidence," *New directions for evaluation*, vol. 1996, no. 70, pp. 29–44, 1996.

[19] M.-P. Dubuisson and A. K. Jain, "A modified hausdorff distance for object matching," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision &amp; Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1. IEEE, 1994, pp. 566–568.

[20] P. Senin, "Dynamic time warping algorithm review," *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, pp. 1–23, 2008.