Motion Synopsis for Robot Arm Trajectories

Daniel Rakita, Bilge Mutlu, and Michael Gleicher

***Abstract*— Monitoring, analyzing, or comparing the motions of a robot can be a critical activity but a tedious and inefficient one in research settings and practical applications. In this paper, we present an approach we call *motion synopsis* for providing users with a global view of a robot's motion trajectory as a set of key poses in a static 2D image, allowing for more efficient robot motion review, preview, analysis, and comparisons. To accomplish this presentation, we construct a 3D scene, select a camera view direction and position based on the motion data, decide what interior poses should be shown based on robot motion features, and organize the robot mesh models and graphical information in a way that provides the user with an at-a-glance view of the motion. Through examples and a user study, we document how our approach performs against alternative summarization techniques and highlight where the approach offers benefit and where it is limited.**

I. INTRODUCTION

Understanding robot motion is important in many scenarios. Tasks such as reviewing what a robot has learned during programming by demonstration, analyzing the output of a motion optimizer, or previewing a motion generated by a motion planner to mitigate mistakes and safety hazards all require an understanding of robot motions. Describing particular motion qualities may be difficult, as the user must vigilantly attend to the whole motion; for example, questions may include "Did the robot maintain an upright end effector throughout the motion," "Will the robot collide with anything in the scene if this planned motion is executed," "In which of these four motions will the robot push the blue button but not the red button," "Did the robot pack all of the correct parts into a box prior to shipment," or "What state is the robot in about halfway through this trajectory."

While observing a robot perform the motion may work in certain scenarios, this approach can be inefficient or infeasible. Analyzing the motion on a physical robot may be time consuming and may lead to wear and tear on the robot over time. It requires the user to have the understanding to match observable aspects of motion to the program controlling the robot. In comparison tasks, watching simulated robot motions is a serial process, each time instance appearing individually in sequence. Thus, understanding all of the qualities of the motion requires steady attention and working memory throughout the motion's duration. Comparing multiple motions compounds these issues as each motion must be attended to in series, requiring more attention, memory, and time for users. Speeding up motion playback may lead to higher error rates and decreased motion understanding.

D. Rakita, B. Mutlu, and M. Gleicher are with the Department of Computer Sciences, University of Wisconsin–Madison, Madison, WI 53706, rakita@cs.wisc.edu, bilge@cs.wisc.edu, gleicher@cs.wisc.edu. This work was supported by National Science Foundation Award # 1208632.

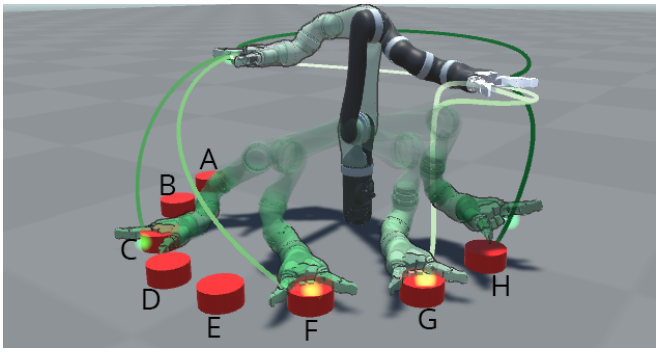

Fig. 1. Our motion synopsis approach summarizes a robot's motion in a static 2D image. We construct a three-dimensional scene, select a camera viewing direction and position in this scene based on the motion data, extract important poses to aptly summarize the motion, and organize the robot mesh models and graphical information in the image such that a user can understand the motion at a glance.

A potential way to address these drawbacks is to *summarize* the robot's motion in a static image or diagram. Possible strategies are to provide glyphs or symbols to semantically indicate motion over time or portray a 3D representation of the robot's motion. While such a system could afford easier and more efficient motion analyses and comparisons, there are numerous challenges to consider. First, robot motion involves a large number of features, including end-effector path, end-effector orientation, joint-velocity information or acceleration information, joint-torque information, the convex hull of the robot sweep, and full robot-state representations. Deciding on an appropriate subset of features to show is a key challenge for a motion-synopsis system. Second, summaries must highlight key features using the most informative and effective representations. Should the graphical elements look like robots in the image, or would spline-based or abstract shape elements allow for more efficient motion exploration? Should there be any text or graph overlays to support the synopsis elements? Finally, to support efficient analysis of robot motions, we must consider how the graphical information should be organized and rendered. For instance, what color or opacity choices afford easy exploration of robot motion? How should collisions among graphical elements be avoided or handled?

In this paper, we propose *robot motion synopsis*, a solution that addresses challenges of summarizing robot motion (an example synopsis image generated by our system can be seen in Figure 1). Our approach starts with a three-dimensional scene, selects a camera viewing direction and position based

on the motion data, extracts important poses to aptly summarize the motion, and organizes the robot mesh models and graphical information in a way that provides the user with an at-a-glance view of the motion (Section III). We demonstrate the use of our approach in several analysis tasks and evaluate user performance with it against alternative forms of motion summarization in a user study (Section IV). The specific contributions of our work include:

- Introducing the idea of *robot motion synopsis* and highlighting tasks where it could be useful;
- Outlining the *design space* for creating effective motion synopses for robotics applications;
- Describing a *prototype implementation* of our approach and assessing its performance in motion review tasks.

II. RELATED WORK

While neither the problem of summarizing robot motion nor potential solutions have been investigated by prior work in robotics, ideas related to motion synopsis have been explored in cinematography, computer animation, video summarization, and information visualization. For a thorough account of work representing motion in images, refer to the survey by Cutting [1]. Early explorations of showing motion in pictures date back to camera pioneers experimenting using various cameras at fixed exposure times, a technique called *chronophotography* [2] (see Marta [3] for a thorough historical account).

In animation, Assa et al. [4] have explored summarizing general animations in an image. Their approach uses affinity matrices to detect poses that are most disparate from each other to serve as key poses. They then take figures in these key poses and use heuristics to strategically place them in the frame, accompanied by motion curves and glyphs. While this approach is informative, robot motion presents unique challenges not generally present in animation with articulated characters. First, robotics applications must support efficient reviewing of summaries. Second, many occlusions are likely in robot arm motion because a robot's base is stationed in one distinct location throughout the trajectory, and additional poses are likely to cover each other when they emanate from this static location. While occlusions are possible in character animation, they are less likely and easier to deal with since characters generally move around in a global space.

Methods in video summarization generally take a video and produce a set of images, shortened video clips, graphical representation, or text to more succinctly describe the video. These methods often require keyframe extraction, a process where the system decides what sequence of frames best describe the video content (see Ajmal [5] and Money and Agius [6] for an overview of video summarization techniques and concepts). Prior work shows summarizing video content in a panoramic image to effectively support review tasks [7]–[11]. For instance, Ponto et al. [12] present an approach for summarizing a user's virtual reality experience as a sequence of images. Their approach then regenerates the egocentric playback view using these extracted viewpoints to create a smoother and clearer camera path.

In information visualization, prior work has explored how to aptly display motion in a static image using visual encoding principles. For example, Bryden et al. [13] use normal mode analysis (NMA) to create illustrations of molecular flexibility. Like robot motion, molecular motion is a complex, high dimensional process. Ware et al. [14] visualize a humpback whale's motion path over the course of a few hours in a static image. Finally, Schroeder et al. [15] present a way to compare differences between thousands of similar biomechanical motions under different conditions.

Informed by this body of work, design requirements for our approach include (1) reducing high dimensionality of robot motions and featuring robot poses based on an importance metric, (2) displaying a global motion over a relatively long period of time, allowing a user to understand a motion at a glance instead of watching back all time frames in succession, and (3) making comparisons easier by providing multiple adjacent synopsis views. The next section describes how we address these requirements.

III. APPROACH OVERVIEW

Our approach uses properties from a robot motion to automatically generate a synopsis image. We construct a full 3D scene because our output image serves as a full representation of how the motion would look in a 3D space. In this scene, we display full 3D mesh models to represent what the robot was doing at particular times through the trajectory. We place models at multiple time points in the scene to show the user interesting robot states throughout the motion. To do this, we must extract what pose configurations are important enough to be shown in the image. We note that placing full pose configurations often creates collisions in the image space, so we allow poses to be broken up into modular link configurations. We observe that even if the end effector is the only link to appear on the path at time t, the user can still discern the end effector's position and orientation information, perhaps indicating what the robot is doing at that time while avoiding pose collisions.

In this section, we provide the general ideas of our approach and walk through necessary implementation details.

A. CAMERA FACING DIRECTION AND POSITION

A view transformation is a particular coordinate system such that points in space can be described with respect to this frame. This derived coordinate system can be placed in a desired position with respect to a global frame by using a translation. Together, this rotation and translation of frame complete the affine transformation mapping vectors to this new descriptive frame of reference.

Viewing the end effector trace best fit plane perpendicularly will clearly show the full sweep of the motion, providing an idea of what actions the robot executed. We find this best fit plane using singular value decomposition (SVD). Viewing the joints from an orthogonal point of view shows how each joint contributed to the final trajectory. Lastly, taking a forward direction into account provides some intuition of what direction to view the robot from and makes

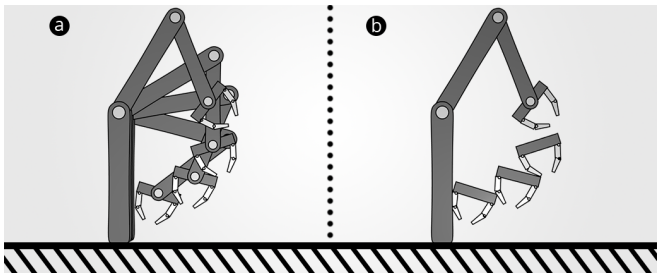

Fig. 2. (a) Because robot links all emanate from the same root point throughout the trajectory, there is a high chance of poses colliding. (b) Having the ability to omit some posterior links in the chain makes the image more legible and always provides at least the end effector position and orientation.

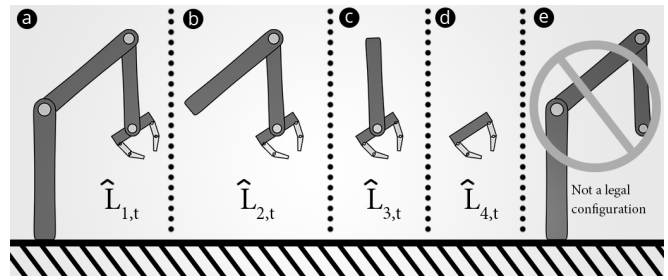

Fig. 3. Our approach allows for posterior joints to be omitted from the chain. However, we always ensure that all joints forward in the chain are present with respect to the back-most joint. This creates $\bar{l}$ legal configurations, $\bar{l}$ being the number of links in the chain, shown here in (a) - (d). (e) depicts an illegal configuration because the end effector link is missing while links behind it in the chain are present.

motions easier to compare. In automatically generating a camera view direction, the goal is to align the camera's forward vector as best as possible with the three vectors that represent these three criteria.

The facing direction for the camera is found using a non-linear optimization formulation. The goal of this optimization is to align the camera's forward vector as best as possible with three vectors: a vector that is the perpendicular end effector path point best fit plane ($\vec{c}$); a vector that serves as a representative of all the joint rotation normals throughout the motion; and lastly, a canonical forward vector defined by the user ($\vec{f}$). We consider the vectors orthogonal to all n joints at every time step t in the trajectory. This criterion does not apply to all joint types as it assumes all planar, revolute joints. A metric that characterizes the motion contribution from non-planar or prismatic joints is left to future work. We denote the ith joint normal, $1 \leq i \leq n$, at time t as $\vec{j}_{i,t}$.

The optimization formulation is as follows:

$$\min_{x} -[\alpha(x\cdot\vec{c})^2 + \beta(x\cdot\vec{f})^2 + \gamma(\frac{1}{t_{max}\cdot n}\sum_{t=1}^{t_{max}}\sum_{i=1}^{n}(x\cdot\vec{j}_{i,t}))^2], \quad (1)$$
$$s.t. \quad ||x|| = 1, \quad x\cdot[0\ 0\ 1]^T \geq 0.$$

α, β and γ are weights for various terms. The first constraint ensures that the magnitude of the facing direction remains one. This bounds the feasible space, ensuring that the dot products do not go to infinity. The second constraint bounds the solution to the hemisphere in front of the robot, preventing the camera from viewing the robot from behind. The solution is found using an interior point solver, and on average it completes in under 2 seconds.

Once a camera-facing direction is found, we can place the camera in world space using a translation. We ensure that the camera is an appropriate distance away, such that the whole motion can be seen. We then align the base and the robot model with the vertical thirds of the image plane to make a more aesthetically pleasing composition. This adheres to the well known "rule of thirds" from photography and cinematography [16].

B. POSE EXTRACTION

In our approach, additional robot models appear in supplementary poses along the trajectory to encode motion over time. These poses must be carefully chosen from the function $\xi(t), t \in (0, t_{max})$ to legibly convey the motion. The robot base does not move, so there is great potential for occlusion between poses. Thus, a good solution is one that balances the tradeoffs between portraying enough additional poses to view novel parts of the motion, without adding an excess number of poses that clutter the image. We introduce a greedy search algorithm that traverses a space of possible poses given a pose importance metric. This pose metric places most consequence on collisions in screen space and world space, and rewards interesting robot motion features such as anomalous joint rotation values, joint angular velocity extrema, and end effector path extrema.

1) Pose Configurations: Because robot links all emanate from the same static base point, there is a great chance that occlusions will occur around that point for all extracted poses. Given a particular pose, we note that even if some links closer to the robot base are omitted, there is still information that can be garnered from more forward links. For instance, even if the end effector is the only link to appear on the path at time t, the user can still learn the end effector's position and orientation at time t, perhaps indicating what the robot is doing at that time while avoiding pose collisions. A 2D illustration of this effect can be seen in Figure 2. For this reason, instead of extracting motion elements at the level of full poses, we instead break up poses into its $\bar{l}$ modular links. We always enforce that extracted poses contain all links ahead in the chain with respect to the back-most link because there is inherently more information to be gained from the front links in the chain. This can be seen illustrated in Figure 3. We refer to a specific pose configuration at time t as $\hat{L}_{i,t}$, $i \in \{1, 2, ..., \bar{l}\}$. In Figure 3, $\bar{l} = 4$. At each time t, our approach can choose to display $\hat{L}_{1,t}, \hat{L}_{2,t}, ..., \hat{L}_{\bar{l},t}$, or no pose configuration. Thus, the number of possible configurations for our approach is $(\bar{l}+1)^t$.

2) Robot Feature Vectors: Before we can decide what pose configurations are important enough to show in the synopsis image, we must decide what motion features are more or less important. With this information, our approach scores these features based on the motion data, stores these scores in robot feature vectors, and searches a space of pose

Algorithm 1 This procedure bootstraps the search process by finding a legal pose configuration with the highest score.

```
procedure FINDFIRSTPOSE
    FirstPose ← [ ]
    for i ← 1 : J̄ do
        scores ← [ ]
        for t ← 1 : t_max do
            currScore = S(L̂_{i,t})
            scores.add(currScore)
        [scores, indices] ← sort(scores)
        if scores.pop ≠ 0 then
            p ← indices.pop
            FirstPose ← L̂_{i,p}
            return FirstPose
```

configurations based on these robot feature vectors using a pose importance metric.

There are many candidate robot characteristics that could be used in such an importance metric. Some examples include extrema in joint angular velocities or accelerations, extrema in the end effector path in Cartesian space, curvature of the end effector path, end effector proximity to objects in the scene, or extrema in energy signals relating to joint torques. In our implementation, we use anomalous joint rotation values, joint angular velocity extrema, and end effector path extrema as features.

a) *Anomalous Rotation:* Using the anomalous rotation feature, the pose importance metric will reward poses that diverge from the expected rotation values for a certain joint. These types of poses should be present in the synopsis as they may be difficult to infer from other poses in context, and they may highlight noteworthy motion qualities. For instance, perhaps a robot unexpectedly swings its elbow joint up while performing an optimized trajectory. The user would not be aware that the motion optimizer caused this result if this anomalous pose was not present in the summary. We will refer to this feature as $\mu(k)$. We compute $\mu(k)$ by taking the distance between the rotation value at time t for joint j and the mean rotation value for joint j throughout the whole trajectory. These distances are normalized from $0.0-1.0$ for each joint.

b) *Joint Angular Velocity Extrema:* Times where a particular joint is at an angular velocity minimum or maximum signify some change in action. Perhaps the robot is ready to pick something up or is changing directions to go towards a new target. Because these poses carry important information about the trajectory, we select this as a feature vector for our importance metric. We refer to this feature as $\nu(k)$. Each element in $\nu(k)$ is found by locating the peaks in each joint angular velocity signal, and a score is assigned based on each point's distance to a closest peak. The score is inverted and normalized such that being closer to a peak gets a higher score between $0.0-1.0$.

c) *End Effector Path Extrema:* Lastly, end effector path extrema likely signify points where a robot performs an action. For example, if a robot pushes a button, the position

Algorithm 2 This procdure takes the output of `FindFirstPose` as a starting configuration group, and greedily attempts to include new legal poses that improve the aggregate pose importance score. The output is a group of poses that aptly summarizes a robot motion.

```
procedure POSEIMPORTANCE SEARCH
    bestPoseGroup ← FindFirstPose
    for i ← 1 : J̄ do
        scores ← [ ]
        for t ← 1 : t_max do
            currScore = S(L̂_{i,t})
            scores.add(currScore)
        [scores, indices] ← sort(scores)
        while scores.pop ≠ 0 do
            bpgScore = S(bestPoseGroup)
            p ← indices.pop
            candidateGroup ← L̂_{i,p} + bestPoseGroup
            candidateGroup.removeCollisionLinks
            candidateGroup.removeIllegalLinks
            cgScore = S(candidateGroup)
            if cgScore > bpgScore then
                bestPoseGroup ← candidateGroup
    return bestPoseGroup
```

where the button is pressed will create a sharp point in the end effector path, which will appear as an extremum in the path curve. We refer to this feature as $\zeta(k)$. We take the end effector points$\{e_1, e_2, ..., e_k\} \subset \mathbb{R}^3$ computed using the forward kinematics of the robot and find extrema points in this 3-dimensional curve. Each value in $\zeta(k)$ is computed by finding the distance to a nearest peak along this curve, and again normalized.

Because our approach assesses modular link configurations at each time frame t, we must have a way to analyze the individual robot joints at every time step. We store all of our robot feature scores in $(\bar{J} \times t_{max})$—length vectors, $\bar{J}$ referring to the number of joints in the robot arm. We access elements in these vectors in what we call time-joint indexing (TJI). The vector consists of a sequence of $\bar{J}$ sized blocks, each block corresponding to a single time t. Each value in this block corresponds to a joint in the robot chain, stored in increasing order from root to end effector.

3) Search Algorithm: Using the robot motion features described above, we now define a pose importance metric and greedy search algorithm that seeks plausible poses to summarize a motion. The scoring metric rewards poses that score highly in the robot feature vectors, and strongly penalizes pose configuration groups that contain collisions in world space or screen space. This aligns with our goal of showing as many novel pose configurations as possible without cluttering the image with colliding poses. The metric takes in a set of TJI values representing a set of current poses, and returns a score based on how well these poses summarize the motion. We will refer to a set of candidate TJI values as K. We enforce that all poses ready to be scored are proper

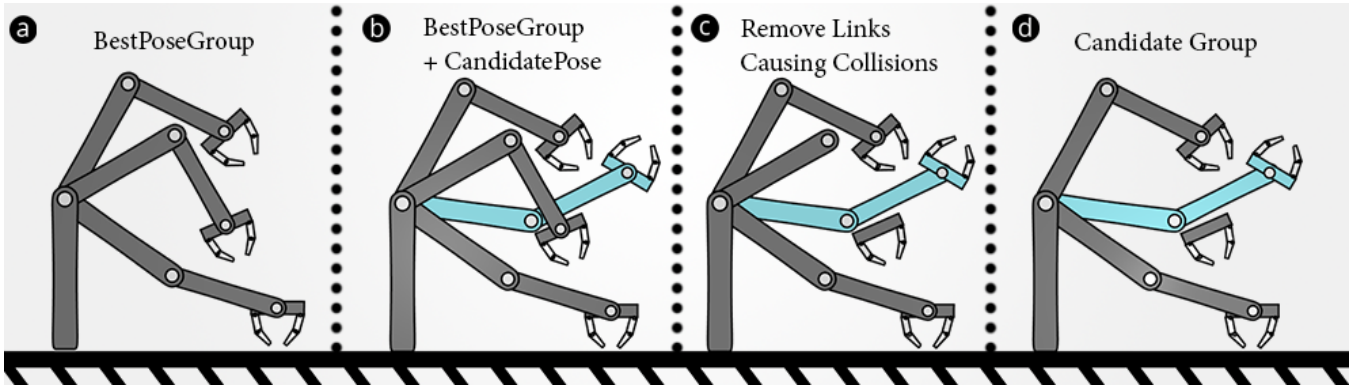

Fig. 4. Illustration of the `PoseImportanceSearch` algorithm. (a) We start with the current best pose group found, (b) greedily find a new candidate pose, (c) remove any links in the pose group that collide with the new candidate pose, and (d) make any additional link modifications to ensure all legal pose configurations as dictated in Figure 3. The score of this candidate group is then compared to the score of the previous best pose group.

pose configurations, *i.e.* all TJI indices are accompanied by all joints ahead in the chain at its time frame, as illustrated in Figure 3.

Our scoring metric is as follows:

$$S(K) = \sum_{k \in K} (\alpha\mu(k) + \beta\nu(k) + \gamma\zeta(k))\hat{C}$$
$$\hat{C} = \prod_{f \in CP} C_s(f,k)C_w(f,k) \prod_{q \in K, q \neq k} C_s(q,k)C_w(q,k) \tag{2}$$

Here, α, β, and γ are all weights for various objectives. The $\hat{C}$ term accounts for all collisions in screen space and world space of the current poses in K. The C_s and C_w functions discern whether a collision is present in screen space or world space, respectively. These functions return 0 if a collision is present, and 1 if there is not. Thus, if any collision is detected between any two links in either space, the whole score is zeroed out. We allow the collision terms to zero out the whole score because we place a hard constraint on collision to maintain a legible summary image. These functions approximate collisions using the Cartesian and screen points calculated at each joint point during the FK preprocessing step. Euclidean distance thresholds are provided for each joint in the chain to indicate whether a collision is present or not. The set CP in equation 2, standing for constant poses, is a set of pose TJI values that will always be seen in the image, and should always be considered in collision calculations. In our system, the full poses corresponding to $\xi(0)$ and $\xi(t_{max})$ are always present and are added to CP.

Our pose importance search first finds a plausible starting pose configuration by greedily selecting the longest pose configuration that does not collide with the starting and ending poses. This is outlined in Algorithm 1. Using this configuration as a starting point, our approach then finds the poses of chain length $\bar{l}$ with the highest scores. The system tries adding in each new pose individually and checks if the aggregate group score is higher with the new pose added. Prior to scoring the new candidate group, we ensure that any links colliding with our new candidate pose are removed and delete any necessary links to ensure all legal pose configurations. This can be seen in Figure 4. If the new group scores higher with the candidate pose added and modifications made, this candidate group becomes our new best pose group. This process continous with robot chains of length $\bar{l}-1, \bar{l}-2, ..., 1$. A full algorithmic description of this search can be found in Algorithm 2.

C. COLOR ENCODING

Once the additional robot models $r_1 \ldots r_j$ are posed in the scene, we decide what color to make each model. If each robot were naïvely left as their native color, the separate models would be difficult to distinguish and the trajectory would be illegible. We index a color from a ColorBrewer sequential color ramp [17] using the arclength parameterized time along the trajectory. The color ramp used can be seen in Figure 5 below. If the time value is between colors, the colors are simply linearly interpolated. In this way, color is used to encode the passage of time, conveying motion in a static image. This is helpful when the robot arm passes through similar points in a long trajectory, helping the viewer understand the relative times the robot was at various locations. Although posterior joints can be omitted by our system to clear the view space, we observe that having floating links in space can be jarring for a user. Thus, we instead smoothly fade out all posterior joints that our system decided to omit using an opacity gradient. We calculate the distance from the most forward point along the length of all omitted links, and pass these values to the vertex shader using a texture map. This effect can be seen in Figure 1.

IV. EVALUATION

To assess the effectiveness of our motion synopsis approach, we conducted a user study to compare motion

Fig. 5. Sequential color ramp used to convey passage of time through the motion trajectory

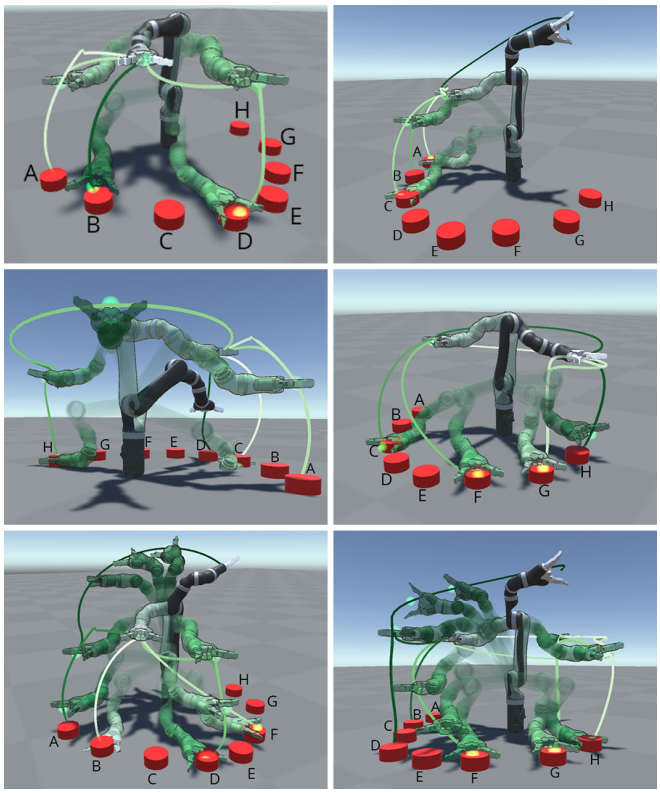

Fig. 6. Examples of motions generated by our motion synopsis system. As part of a motion review task, study participants were asked to specify which of the eight labeled buttons the robot pushed throughout the motion.

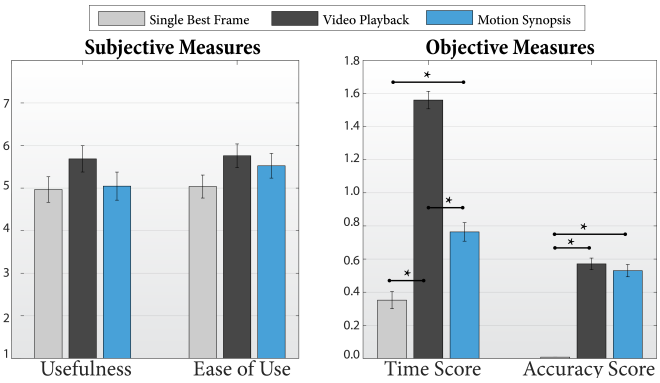

Fig. 7. Results from our human-subjects study. We adapted two scales by Davis [18] on *ease of use* and *usefulness* to human-robot interaction applications. Time score is the aggregate task time normalized by motion length. For accuracy, we only considered answers correct if all button selections were correct. *Error bars represent standard error.* * *denotes* $p \leq .01$, † *denotes* $p \leq .05$

description accuracy and task performance against motion summarization alternatives. In this section, we outline our experimental setup and results.

A. METHOD

The participant study followed a 3 × 1 (*summary type*: motion synopsis image, video playback 4× speed, and best frame) between participants design. The *video playback at 4× speed* serves as a likely tool currently used for motion summarization in practice. The *best frame* condition selects a single frame that illustrates a pertinent part of the motion, serving as a naïve motion synopsis alternative. In our experiment, *best frame* images portrayed the robot arm pushing a single button. We recruited 60 participants (30 male, 30 female), 20 per condition, using Amazon Mechanical Turk. Participant ages averaged 33.83 (*SD*=11.35). Only one participant noted prior or current experience with robots. No participants reported participating in prior robotics research studies. The study took five minutes to complete, and each participant received 50¢ USD for his/her time.

The participants were shown six scenes in succession, each presented using one of the summary methods. Each scene contained eight buttons labeled "A" through "H," and the task was to select which buttons the robot has pushed throughout the motion. The robot could push one or many buttons, and the participant was asked to select all that apply. Our motions consisted of 3–5 button pushes and ranged in time between 30s–93s. The motion synopsis images portraying the six motions can be seen in Figure 6. The participant was given unbounded time to complete each task. After all six motions, the participant was presented with a questionnaire regarding the summary type. In constructing our subjective scales for post-experiment questionnaires, we adapted two scales by Davis [18] on *ease of use* and *usefulness* to human-robot interaction applications. Objective measures included task time and accuracy. For task accuracy, answers were only considered correct if all button push responses were correct. Task time was normalized by the length of the motion.

B. HYPOTHESES

H1: We hypothesize that participants will be able to answer faster using our motion synopsis than video playback. We make this prediction because a synopsis can be read at a glance, rather than requiring watching the video over time.

H2: We hypothesize that motion synopsis will perform better than video on accuracy and subjective scores (ease of use and usefulness scales).

H3: We expect both motion synopsis and video will outperform best frame on accuracy. The basis of this prediction is that synopsis presents all relevant motion features for easy access, rather than relying on the viewer's memory.

C. RESULTS

A summary of our results can be seen in Figure 7. Participant data was analyzed using a one-way analysis of variance (ANOVA). All pairwise comparisons were made using Tukey HSD Post Hoc tests. H1 stated that participants would be able to answer faster using our motion synopsis over video playback. Participants did answer faster ($p < .001$) using our motion synopsis approach ($M = 0.76$, $SD = 0.24$) than video summary ($M = 1.56$, $SD = 0.33$). H2 stated that motion synopsis will perform better than video on accuracy. Our analysis found no differences, failing to provide support for H2. H3 stated that both video and motion synopsis would outperform best frame. The data reflects this ($p < .001$

between motion synopsis and best frame, $p < .001$ between video and best frame) with a best frame average accuracy of 0.0075 ($SD = 0.04$). Lastly, H2 states that motion synopsis would perform better than video on the ease of use and usefulness subjective measures. We found no significant differences between video and motion synopsis on either of these scales.

D. DISCUSSION

Our data partially support our hypotheses; the motion synopsis approach provides significant improvement over video playback in task time while not compromising from accuracy. While our approach does not provide a task accuracy advantage, we show that it leads to similar motion-descriptiveness benefits as video in about half the time (measured in task time), making it a practical solution for motion summarization tasks. We believe that this time advantage will compound when comparing multiple motions in parallel. We plan to assess this effect in future work.

It is possible that the synopsis approach is novel to most participants, and effective use of synopses requires some experience. To achieve higher accuracy, synopses may require training or may be better suited to expert use.

While our synopsis approach provided task benefits, it was not found by participants to be easier to use or more useful than the baseline methods. We speculate that this finding is because participants could not compare the different conditions themselves in the between subjects design, and the novelty of performing a robot related task resulted in overall heightened ratings.

V. GENERAL DISCUSSION

In this work, we have presented the problem of understanding robot motions in an efficient and effective manner, and we provide a motion synopsis approach to automatically create static images that provides an at-a-glance view of the motion. Our approach is shown to be effective by affording comparable motion description accuracy to a video playback in half the task time. While we believe that the idea of synopsis has potential, we note that our initial approach has limitations that we hope to address in future work.

In the future, we plan to explore the quality of motion synopsis performance in particular tasks and accompanying motions. For instance, simple motions and tasks may not provide benefit over video. To achieve the potential of the approach, we may need an improved design that is easier to interpret, a better understanding of where the approach has merit, and improved instructions to enable new users to interpret the images. For longer or complex motions, our design may not scale well as the display will quickly become too cluttered. This is a fundamental challenge of a synopsis approach. We will explore solutions to this issue including segmenting the motion to use multi-frame images (such as a comic strip) and allow for interactive exploration. Also, a single design may not be adequate for all motion analysis tasks; we plan to develop specialized designs that are specifically suited for key tasks such as assessing clearance.

VI. CONCLUSION

In this paper, we show that the area of motion synopsis has potential in robotics applications. The problem of understanding robot motions in an efficient and effective manner has been previously unexplored in robotics applications. We provide an initial motion synopsis approach to automatically create static images that provide an at-a-glance view of the motion. Through a user-study, our approach is shown to be a practical alternative to a video playback method for a motion review task. We will continue to expand upon this approach and evaluate its performance against alternative methods for summarizing robot motion in future work.

REFERENCES

[1] J. E. Cutting, "Representing motion in a static image: constraints and parallels in art, science, and popular culture," *Perception*, vol. 31, no. 10, pp. 1165–1193, 2002.

[2] R. Hirsch, "Seizing the light," *A History of Photography, New York*, 2008.

[3] M. Braun, *Picturing time: the work of Etienne-Jules Marey (1830-1904)*. University of Chicago Press, 1994.

[4] J. Assa, Y. Caspi, and D. Cohen-Or, "Action synopsis: pose selection and illustration," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 667–676, 2005.

[5] M. Ajmal, M. H. Ashraf, M. Shakir, Y. Abbas, and F. A. Shah, "Video summarization: techniques and classification," in *Computer Vision and Graphics*. Springer, 2012, pp. 1–13.

[6] A. G. Money and H. Agius, "Video summarisation: A conceptual framework and survey of the state of the art," *Journal of Visual Communication and Image Representation*, vol. 19, no. 2, pp. 121–143, 2008.

[7] M. Irani, P. Anandan, and S. Hsu, "Mosaic based representations of video sequences and their applications," in *Computer Vision, 1995. Proceedings., Fifth International Conference on*. IEEE, 1995, pp. 605–611.

[8] L. Teodosio and W. Bender, "Salient video stills: Content and context preserved," in *Proceedings of the first ACM international conference on Multimedia*. ACM, 1993, pp. 39–46.

[9] R. Szeliski and H.-Y. Shum, "Creating full view panoramic image mosaics and environment maps," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 1997, pp. 251–258.

[10] Y. Caspi, A. Axelrod, Y. Matsushita, and A. Gamliel, "Dynamic stills and clip trailers," *The Visual Computer*, vol. 22, no. 9-11, pp. 642–652, 2006.

[11] F. Liu, Y.-h. Hu, and M. L. Gleicher, "Discovering panoramas in web videos," in *Proceedings of the 16th ACM international conference on Multimedia*. ACM, 2008, pp. 329–338.

[12] K. Ponto, J. Kohlmann, and M. Gleicher, "Effective replays and summarization of virtual experiences," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 4, pp. 607–616, 2012.

[13] A. Bryden, G. N. Phillips Jr, and M. Gleicher, "Automated illustration of molecular flexibility," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 1, pp. 132–145, 2012.

[14] C. Ware, R. Arsenault, M. Plumlee, and D. Wiley, "Visualizing the underwater behavior of humpback whales," *IEEE Computer Graphics and Applications*, no. 4, pp. 14–18, 2006.

[15] D. Schroeder, F. Korsakov, C. M.-P. Knipe, L. Thorson, A. M. Ellingson, D. Nuckley, J. Carlis, and D. F. Keefe, "Trend-centric motion visualization: Designing and applying a new strategy for analyzing scientific motion collections," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 20, no. 12, pp. 2644–2653, 2014.

[16] Q. Galvane, "Automatic cinematography and editing in virtual environments," Ph.D. dissertation, Grenoble 1 UJF-Université Joseph Fourier, 2015.

[17] M. Harrower and C. A. Brewer, "ColorBrewer.org: an online tool for selecting colour schemes for maps," *The Cartographic Journal*, vol. 40, no. 1, pp. 27–37, 2003.

[18] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS Q.*, vol. 13, no. 3, pp. 319–340, 1989.