# *Motion Comparator*: Visual Comparison of Robot Motions

Yeping Wang[1], Alexander Peseckis[1], Zelong Jiang[1], and Michael Gleicher[1]

*Abstract*— **Roboticists compare robot motions for tasks such as parameter tuning, troubleshooting, and deciding between possible motions. However, most existing visualization tools are designed for individual motions and lack the features necessary to facilitate robot motion comparison. In this paper, we utilize a rigorous design framework to develop *Motion Comparator*, a web-based tool that facilitates the comprehension, comparison, and communication of robot motions. Our design process identified roboticists' needs, articulated design challenges, and provided corresponding strategies. Motion Comparator includes several key features such as multi-view coordination, quaternion visualization, time warping, and comparative designs. To demonstrate the applications of Motion Comparator, we discuss four case studies in which our tool is used for motion selection, troubleshooting, parameter tuning, and motion review.**

## I. Introduction

Roboticists often employ various quantitative metrics such as duration or average accuracy to evaluate and compare motions. While these metrics provide an overall summary of a motion, they are insufficient for roboticists to *understand* the finer details between them. For example, such metrics do not allow roboticists to explain quantitative findings, identify unusual events, or evaluate motions in a subjective or task-specific way such as assessing their predictability, legibility, or social acceptance. Roboticists need to acquire a comprehensive understanding of each motion before they can decide which motion to deploy. Meanwhile, the act of comparing also facilitates the understanding of each individual motion.

In this paper, we introduce *Motion Comparator*, a web-based system that facilitates the comprehension, comparison, and communication of robot motions. It provides users with different visualizations of robot motions, each designed to help with motion understanding and comparison. The system supports flexible layout and view coordination to allow users to create workflows that support their needs.

We design our system by applying a rigorous visual comparison design framework [1] which involves 1) identifying user *needs*, 2) analyzing the comparative *challenges* and *strategies*, and 3) considering the visual *designs* for comparison. First, we identify seven motion properties that are of interest to roboticists through a systematic examination of the abstracts of papers published at major venues. Then we identify three major *challenges* that make robot motions difficult: temporal length of robot motions, spatial complexity of robot motions, and complexity of relationships between motions. To assist the visualization of temporally long motions, Motion Comparator unfolds time into space by showing traces in Cartesian space, quaternion space, and joint space. Furthermore, Motion Comparator allows users to select, visualize, and compare motion segments. In order to address spatial complexity, Motion Comparator provides multiple coordinated views that allow users to simultaneously inspect a motion using various visualization methods. To mitigate the complex relationship between motions, Motion Comparator utilizes time warping to automatically align two motions. Motion comparitor employs multiple views based on the basic comparative designs (juxtaposition, superposition, and explicit encoding) in order to satisfy various comparison needs. The key features of our system are summarized in Table I. We have open-sourced Motion Comparator[1] and invite readers to use our web-based system[2].

The main contribution of this paper is a visualization tool for roboticists to understand, compare, and share robot motions. In doing this, we make specific contributions including: (1) we provide a literature-based requirements analysis to articulate the visualization needs of robot motions. The analysis informs future tool design and reveals the diversity of these needs, suggesting that a single visual design is unlikely to meet all requirements. (2) We demonstrate how various visualization approaches, such as traces in different spaces and quaternion visualization, can be adapted to motion comparison problems, providing coverage for the range of user needs. (3) We show how multiple visualizations can be integrated by adapting multi-view coordination for motion comparison by applying time-warp based alignment and view synchronization.

In the remainder of the paper, we describe how we designed Motion Comparator in §IV. In §V, we describe Motion Comparator's multi-view interface, where each view is specialized to visualize a specific aspect of robot motions. The implementation details are provided in §VI. Being able to directly parse `rosbag` files, Motion Comparator is a convenient tool for Robot Operating System (ROS) users. In §VII, we provide four case studies in which Motion Comparator is utilized for motion selection, troubleshooting, parameter tuning, and motion review.

## II. Related Works

### A. Robot Visualization Tools

Most existing robot visualization tools focus on single motions. For example, Rviz, a built-in tool in Robot Operating System (ROS), visualizes robot states in a simulated 3D world. MoveIt Visual Tool builds upon Rviz to visualize motions generated by the MoveIt motion planner. The tool animates how a robot executes a motion and draws a trace

[1] All authors are with the Department of Computer Sciences, University of Wisconsin-Madison, Madison, WI 53706, USA
[yeping|gleicher]@cs.wisc.edu
[peseckis|zjiang287]@wisc.edu

This work was supported by National Science Foundation award 2007436, Los Alamos National Laboratory and the Department of Energy.

[1] Source code: https://github.com/uwgraphics/MotionComparator

[2] Link to our tool: https://pages.graphics.cs.wisc.edu/MotionComparator

TABLE I
KEY FEATURES OF MOTION COMPARATOR IN CONTRAST TO EXISTING TOOLS

| Name | Motion Visualization | | | | | | | Motion Comparison | |
|---|---|---|---|---|---|---|---|---|---|
| | 3D Scene | Time Series Plots | Scrubbable Timeline Bar | Shareable | Position Trace | Quaternion Trace | Joint Trace | Time Warping | Comparative Designs |
| Rviz (MoveIt Visual Tools) | ✓ | | | | ✓ | | | | |
| rqt_plot (rqt_multiplot) | | ✓ | | | | | | | |
| robot-log-visualizer | ✓ | ✓ | ✓ | | | | | | |
| PlotJuggler | | ✓ | ✓ | ✓ | | | | | |
| Webviz (Foxglove Studio) | ✓ | ✓ | ✓ | ✓ | | | | | |
| Klamp't | ✓ | ✓ | | | ✓ | | | | |
| Robowflex | ✓ | | | | | | | | |
| Motion Comparator | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

to represent the end-effector path. Aside from them, the ROS ecosystem offers 2D graph tools such as rqt_plot or rqt_multiplot to visualize scalar values such as joint velocities. Unlike these tools that focus on real-time visualization, robot-log-visualizer places emphasis on offline data visualization and offers a scrubbable timeline bar that enables users to navigate through a motion. PlotJuggler is an emerging time-series data plotting tool with a scrubbable timeline bar and some built-in functionalities such as computing derivatives and integrals. In addition, PlotJuggler offers users the functionality to customize, store, and retrieve layouts, which is crucial for communication and collaboration. To further facilitate collaborative efforts, Webviz and Foxglove Studio have been developed as web-based applications that allow users to visualize data by simply opening a browser window and navigating to a URL. In addition to these specialized visualization tools, several general toolboxes, such as Klamp't and Robowflex [2] also incorporate features to visualize robot states in a 3D environment. The key features of all the aforementioned tools are outlined in Table I in comparison to Motion Comparator. These tools are designed for robot motion visualization and do *not* provide features that facilitate visual comparison of robot motions. Meanwhile, Motion Comparator is built from the ground up to compare robot motions.

### B. Motion Visualization

In addition to the robot visualization tools, Motion Comparator also draws inspiration from literature on general motion visualization. Coffey *et al.* [3] provides a taxonomy of motion visualizations that dissects motions by their space and time dimensions. Informed by their work, Motion Comparator offers functionalities that assist users in comparing the space and time dimensions of robot motions. We also adapt the standard visualization approach that uses a trace to depict an object's movements. Position traces have been employed in the visualization of human motions [4], [5], animal movements [6], and the motions of the tooltip of a surgical robot [7]. These traces have been enhanced with annotation to augment their informational content, such as using chevrons to encode directions [7], [6], triangles to encode velocities [8], and color to encode grip force applied by a robot end-effector [7].

While an object's position change can be visualized using a position trace, it is more challenging to visualize its orientation over time. Several works seek to visualize orientations on position traces using color [9] or coordinate frames [10], [11], which can be unintuitive or visually cluttered. An alternative body of literature represents orientations in quaternions and seeks ways to visualize quaternion traces, *e.g.*, by projecting quaternions to a 3D space [12]. Motion Comparator offers both position traces and quaternion traces to visualize the position and orientation changes of an object, *e.g.*, a robot manipulator's end-effector.

Meanwhile, a thread of works focuses on illustrating robot motions using *static* 2D images [13], [14], [15], which can be readily inserted into academic publications to facilitate scientific communication. While Motion Comparator is an *interactive* system that incorporates several 2D and 3D visualization approaches, users can take representative screenshots of Motion Comparator and incorporate them into academic papers. All figures in this paper are assembled from one or more screenshots of Motion Comparator.

### C. Motion Visual Comparison

Gleicher [1] identifies three basic designs for visual comparison of general objects: *1) Juxtaposition designs* place objects next to each other and rely on the viewer's memory to make connections between objects. Consequently, subtle differences may not be apparent in juxtaposition designs. *2) Superposition designs* overlay objects in the same place and facilitate the detection of subtle differences. Superposition designs use the viewer's vision for comparison and one of their notable issues is the occlusion or clutter caused by overlapped objects. *3) Explicit encoding designs* compute relationships between objects and provide visual encoding of the relationships. While relieving viewers from the burden of comparison, explicit encodings require the relationships to be known and viewers may experience a loss of contextual information. Explicit encoding is beneficial in measuring or communicating a relationship but may not be as effective in contextualizing or dissecting a relationship. Kim *et al.* [16] distinguish spatial and temporal juxtaposition, calling the latter category interchangeable designs as such designs alternate between items temporally in the same space. In §IV-C, we will further discuss these three designs with their benefits and drawbacks. To satisfy the various needs of robot motion comparison, Motion Comparator is integrated with all juxtaposition, superposition, and explicit encoding designs.

To facilitate comparison, time warping is often used to

temporally align multiple motions [17]. Gong *et al.* [18] use time warping to compare the motion between a teleoperated robot and its human operator. The alignment after time warping can be visualized by connecting corresponding time frames [19], using warping curves [19], or encoding in colors [20]. Motion Comparator also offers time warping functionality and provides built-in loss functions to customize the temporal alignment of two robot motions.

## III. Problem Overview

In the scope of this paper, we define a *robot motion* as a collection of motion data that encodes how a task is performed. A robot motion may contain the movements of multiple robots and relevant objects, *e.g.*, the object being manipulated. The motion of a mobile robot, a drone, or an object, is represented by a trajectory of poses: $\chi : [0, T] \rightarrow SE(3)$. The motion of a high-degree-of-freedom robot manipulator or humanoid robot is encoded using a trajectory of joint states: $\xi : [0, T] \rightarrow \mathbb{R}^n$, where $n$ is the number of joints. Therefore, robot motions are time-varying, high-dimensional data. This paper aims to provide visualization approaches to facilitate the understanding and comparison of a small number of robot motions. The visual comparison of a large number of robot motions presents unique challenges and likely requires different approaches. Visualization approaches include a collection of 2D graphs, 3D shapes, interactive tools, layouts, and mechanisms such as multi-view coordination. With multiple visualizations and flexible layouts, roboticists can create workflows that support their needs.

## IV. Design Process

We develop Motion Comparator by applying a rigorous design framework [1] which identifies roboticist needs, analyzes comparative challenges and strategies, and considers different visual designs. This section describes our design process and the design choices we make.

### A. Comparative Needs

We first identify the properties of robot motions that are of interest to roboticists. Robot motions are compared by their similarities or differences in these properties.

We systematically identified comparative properties of robot motions from the abstracts of 64,893 papers published in major robotics journals (*e.g.*, the International Journal of Robotics Research), and robotics conferences (*e.g.*, Robotics: Science and Systems). The first step was a filtering process to identify sentences that contain "robot motion", "robot trajectory", "robot path", "robot movement", and their respective plural forms. We collected adjectives and adverbs among the identified sentences. After manually removing ambiguous words (*e.g.*, "important") and combining synonyms, we further examined the remaining words in their original sentences to ensure that they were used to describe robot motions.

Our process identified a total of seven properties. We report the properties, along with their representative sources, in Table II. Some of the identified properties are considered *objective*, *i.e.*, these properties can be unambiguously described in mathematics and communicated using certain metrics. Meanwhile, it is more challenging to measure and communicate *subjective* properties because their criteria are set by individuals and may change depending on the contexts. Motion Comparator aims to facilitate the assessment of both objective and subjective properties of robot motions.

The identified properties describe robot motions from different perspectives. Because robot motions are time-varying, high-dimensional data, we seek to *project* robot motions to some lower dimensional space to reveal the identified properties. As shown in Table II, the identified properties describe motions in three spaces: the Cartesian space which encompasses both robots and humans, the joint space which unambiguously describes a robot configuration, and time which demotes the duration of motions. In order to cover all properties in the spaces, Motion Comparator provides 3D scenes, time-series plot, and timeline bar to display motions in these spaces. We will describe these views in §V.

### B. Comparative Challenges and Strategies

Gleicher [1] classifies the challenges in visual comparison into three categories: the large number of items, the complexity of individual items, and the complexity of the relationships between items. While robot motion comparison has all these challenges, this paper focuses on the visual comparison of a small number of robot motions, avoiding the first type of challenge. We identify the remaining challenges as the temporal length and spatial complexity of each robot motion, as well as the complex relationships between robot motions. Below, we describe the three challenges, our strategies to address them, and the corresponding features offered by Motion Comparator.

*Challenge 1: Temporal length of robot motions* – Robot motions are time-varying data. 3D scene visualizations only show the robot state at a specific timestamp and users need to use the animated timeline bar to view an entire motion. New features should be provided to effectively handle the temporal length of robot motions.

*Strategy 1.1: Traces in various spaces* – A *trace* is a line that displays the positions of an object as time changes. Traces address the temporal length challenge of robot motions by folding time into space. Depending on the space being displayed, Motion Comparator offers traces in Cartesian, quaternion, and joint spaces. Position traces are lines in Cartesian space that depict an object's *position* changes over time. Quaternion traces are lines in 4D Euclidean space that encode an object's *orientation* changes. Joint space traces are lines in a robot's joint space that show how its *joints* move over time. The implementation and visualization of these traces are described in §V-A, §V-B, and §V-D.

*Strategy 1.2: Subset selection* – Another strategy to visualize temporally long motions is to select a subset of the motions. Similar to video editor software, Motion Comparator features a timeline bar that includes a selectable region, allowing users to control the duration of the motion being visualized. This feature enables users to temporally "zoom in" to inspect a particular part of a motion.

TABLE II
MOTION PROPERTIES BEING COMPARED

| | Properties | Cartesian Space | Joint Space | Time | Example Source |
|---|---|---|---|---|---|
| Objective | Efficiency | | | ● | ... other robot trajectories that perform the same task more **efficiently**...[21] |
| | Collision-Free/Safeness | ● | | | ... lead to **safer** and socially more acceptable robot trajectories. [22] |
| | Smoothness/Continuity | ● | ● | ● | ... in terms of faster search, and **smoother** and shorter robot trajectories. [23] |
| | Accuracy | ● | | | ... can be used to automatically generate **accurate** 6 DOF robot paths... [24] |
| Subjective | Predictability | ● | | ● | Robot motion should also be ... when operating among people, **predictable**. [25] |
| | Legibility | ● | | ● | ... thus allowing the generation of smoother and more **legible** robot motion. [26] |
| | Human-Like/Natural | ● | | ● | We demonstrate a method for making robot motion more **human-like**. [27] |

*Challenge 2: Spatial complexity of robot motions* – At each time frame, a robot may move its joints in joint space, resulting in position and orientation changes in Cartesian space. A robot's motion in Cartesian space determines its task performance, *e.g.*, accuracy and collision avoidance. On the other hand, the robot's energy usage, wear, and tear are influenced by its motion in joint space. Hence, it is necessary for roboticists to possess a comprehensive understanding of motions in both Cartesian and joint space.

*Strategy 2: Multi-view coordination* – This strategy aims to assist users in building connections between various visualizations. Our tool provides a scrubbable timeline bar that enables users to examine each frame on robot motions. Multiple visualizations are controlled by the same timeline bar. When users scrub the timeline bar, the 3D scene will visualize the robot and its surrounding environment at the selected time stamp. Meanwhile, the corresponding points on quaternion traces, joint traces, and time-series lines will be highlighted (Fig. 1). In addition, we offer synchronization of views and lighting across views. These features enable users to examine multiple aspects of a robot motion at once.

*Challenge 3: Complexity of relationships* – Robot motions are time-series data that may have different durations, so the direct comparison at each time frame may not be meaningful. For example, when comparing motions for a pick-and-place task, it is more meaningful to compare the robots' poses as they pick up the object, as opposed to their poses at a specific time frame. This example demonstrates the complex relationship between motions, urging comparison tools to seek ways to simplify the comparison.

*Strategy 3: Temporal alignments* – Given that motions are not necessarily frame-to-frame corresponded, our tool facilitates comparison by autonomous temporal alignment. Specifically, we use *time-warping* [28] to establish a correspondence that minimizes some loss function. Motion Comparator offers loss functions as the difference of joint states in joint space, the Euclidean distance in Cartesian space, or a sum of multiple objectives.

### C. Comparative Designs

As a visual comparison tool, Motion Comparator must decide how information is displayed to assist comparison. Prior works [1] have identified three basic categories for comparative designs: juxtaposition, superposition, and explicit encoding. As described in §II-B, these designs have their benefits and drawbacks. Motion Comparator incorporates all these three designs to satisfy various comparison needs. For instance, superposition designs are suitable for detecting subtle differences between robot motions while explicit encoding designs are beneficial for measuring or communicating a relationship. In §V, we will further describe how Motion Comparator was built from the ground up to support comparative designs.

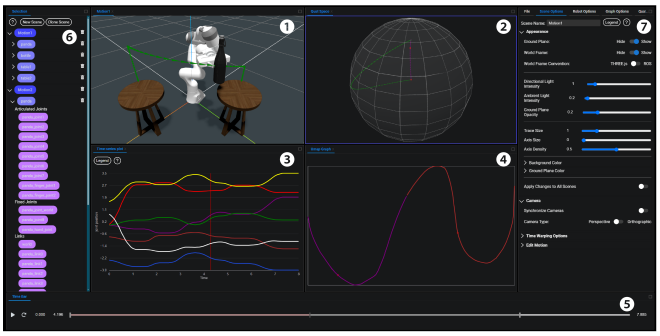


Fig. 1. Screenshot of Motion Comparator showing its multiple panels. Its customizable layout allows users to tailor the interface for various workflows. (1) 3D Scene with a position trace (2) quaternion space (3) time-series plot (4) UMAP graph (5) timeline bar (6) motion library that shows loaded robot motions (7) option panels to change preferences.

## V. MULTI-VIEW INTERFACE

Motion Comparator provides a multi-view interface that visualizes various aspects of motions to satisfy the comparative needs identified in §IV-A. Although certain views may be present in some existing visualization tools, the views in Motion Comparator are built from the ground up to incorporate the strategies in §IV-B and the comparative designs in §IV-C. For example, roboticists can juxtapose two instances of Rviz side-by-side for visual comparison. However, users need to manually synchronize camera viewpoints between the instances and temporally align motions. In contrast, the 3D scenes of Motion Comparator offer features that assist comparison, such as unifying viewpoints and lighting conditions, as well as automatic temporal alignments. Below, we first explain the basic features of each view, then how it incorporates comparative strategies and designs.

### A. 3D Scene

3D scenes in Motion Comparator render robots and their surroundings in simulated 3D environments. To navigate through 3D scenes, users can adjust the camera viewpoint using orbit controls with their mouse. Motion Comparator also offers configurability for customizing lighting and opacity, manually adjusting robot motions, and positioning objects in 3D scenes. 3D scene views are essential for users to see and

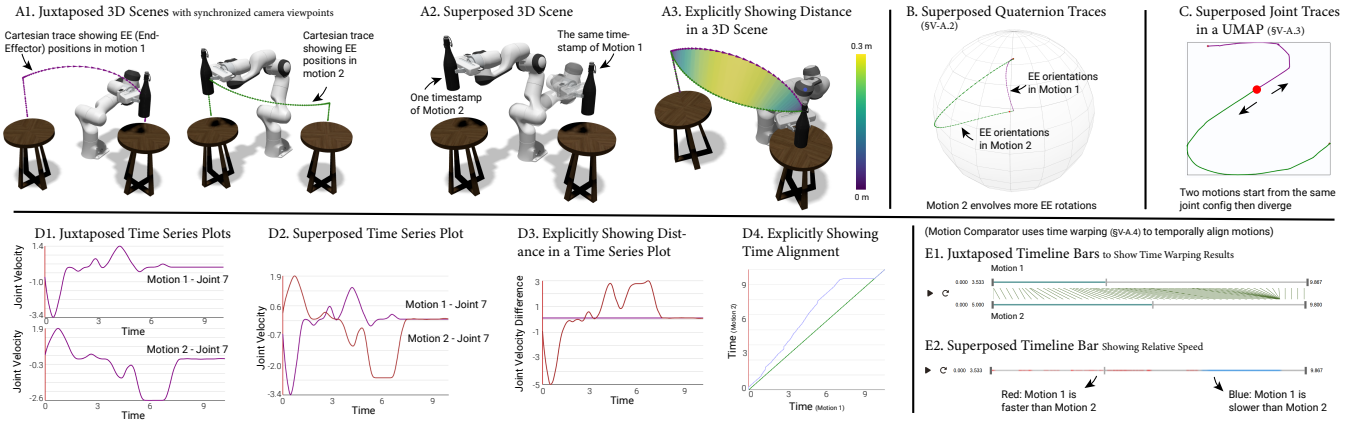


Fig. 2. Motion Comparator offers five views to visualization robot motions, including 3D scene (A), quaternion space (B), UMAP graph (C), time-series plot (D), and timeline bar (E). These views incorporate the comparative strategies in §IV-B, including position trace (A1), quaternion trace (B), joint trace (C), and temporal alignments (D4, E1, E2). These views also incorporate comparative designs, including juxtaposition, superposition, and explicit encoding.

understand how a robot moves. With an animated timeline bar, robot motions can be played in 3D scenes.

As discussed in §IV-B-Strategy 1.1, *position traces* fold time into space to handle temporally long motions. A position trace is a line in a 3D scene that depicts an object's position at each time stamp [6]. As shown in Fig. 2-A1, Motion Comparator draws position traces with consecutive cones to indicate directions and encode velocity.

The customizable layout feature (§VI-3) of Motion Comparator allows users to place multiple 3D scenes side by side, thereby achieving *juxtaposition* designs (see Fig. 2-A1) Motion Comparator can also visualize multiple robot motions in a single 3D scene, thereby achieving *superposition* designs. As shown in Fig. 2-A2, users can change the opacities of robot models and objects to alleviate the problem of occlusion. In addition, Motion Comparator provides another approach that *explicitly* visualizes distances between objects in Cartesian space. As shown in Fig. 2-A3, by filling the gap between Cartesian traces and using color to encode the various distances, Motion Comparator contextualizes the visualization of differences in 3D scenes.

### B. Quaternion Space

As described in §II-B, it is challenging to visualize the orientation change of an object. One approach commonly used in data visualization represents orientations as unit quaternions $\mathbf{q} = (x, y, z, w) \in \mathbb{R}^4$ and visualizes quaternions by projecting them from 4D Euclidean space to a sphere in 3D space [12], [29]. Specifically, we project quaternions to a 3D space where $w = 0$. After projection, each quaternion becomes a 3D vector that is collinear with the rotational axis and its length is correlated with the angles of rotation.

While it may take some training to interpret quaternion traces, they effectively visualize orientation changes [29]. The utility of quaternion traces becomes more apparent when multiple quaternion traces are drawn together. As shown in Fig. 2-B, longer quaternion traces indicate more rotations.

### C. Time-Series Plots

Motion Comparator offers time-series plots to visualize various scalars. These scalars include positions in joint space or Cartesian space and their derivatives (velocities, accelerations, and jerks). To handle noisy data, Motion Comparator provides filters to smooth time-series plots.

Motion Comparator allows users to place multiple time-series plots side by side to make *juxtaposition* designs (Fig. fig. 2-D1). Meanwhile, time-series plots naturally accommodate *superposition* designs because multiple lines can be drawn in a single plot (Fig. 2-D2). To *explicitly* visualize the relationship between two variables, Motion Comparator can visualize their differences in a time-series plot (Fig. 2-D3). It supports difference between robot joint positions, distance between objects, or velocity, acceleration, and jerk difference in Cartesian space or joint space. In addition, Motion Comparator can draw warping curves to *explicitly* show the time alignment between two motions. As shown in Fig. 2-D4, a warping curve is displayed in a 2D plot where both the $x$ and $y$ axes represent time. When a warping curve is above the diagonal reference line, it indicates that the $x$-axis motion is faster than the $y$-axis motion.

### D. Joint Trace

While a time-series plot effectively visualizes the movement of individual joints, the display becomes cluttered when all joints of a robot are shown simultaneously on a single plot. To address this issue, we provide *joint trace*, which leverages dimensionality reduction technologies to depict the movement of *all* joints of a robot in a line. As shown in Fig. 2-C, we visualize joint traces using Uniform Manifold Approximation and Projection (UMAP) [30], a manifold learning technique for dimension reduction. The objective of UMAP is to construct a low-dimensional graph where similar joint states are close to each other and dissimilar joint states are far away from each other. Despite the information loss in dimension reduction, joint trace in UMAP graph is an efficient way for roboticists to quickly identify patterns and gain intuitive insights (§VII-C provides an example). Our current implementation uses the `umap-js` library for dimensionality reduction, which slows down with long motions. To improve performance, we plan to switch to a Python backend using the `umap-learn` library.

### E. Scrubbable Timeline Bar

The timeline bar of Motion Comparator serves the purpose of indicating the duration of motions, animating robot motions, and providing users with a convenient way to navigate through time. In a manner akin to the utilization of timeline bars in video players, users can interact with the timeline bar by either clicking on it to navigate to a specific timestamp or dragging its scrubber to make fine adjustments.

In addition, timeline bars visualize time-warping results and show the temporal alignment between motions. Fig. 2-E1) provides an example of the *juxtaposition* design that compares the temporal characteristics of two motions. When users move the scrubber on one timeline bar, the scrubber on the other timeline bar is automatically moved to the corresponding position. The lines between the timeline bars also illustrate the temporal alignment between motions. Motion Comparator also visualizes time-warping results in a superposition design. As shown in Figure 2-E2, the timeline bar uses color to encode the relative speed between two motions, where blue and red indicate that a motion has a lower and higher speed in comparison to the reference motion. This design is particularly useful for users to dissect the temporal difference between motions. For instance, users can quickly identify that it is the slow pick-up motion that causes a motion to be less efficient than another.

## VI. Implementation

In this section, we describe the implementation details of Motion Comparator, specifically highlighting its role in facilitating *communication* of robot motions.

*1) Web-based interface:* Our goal is to build a web-based interface that is lightweight and cross-platform, eliminating the need to download or install software. Motion Comparator is a web application using `React.js`. It uses `three.js` to implement 3D scenes and quaternion spaces, and `D3.js` for time-series plots and UMAP graphs. Motion Comparator is open-sourced on GitHub, allowing it to benefit from community feedback and facilitating ongoing maintenance.

*2) ROS integration:* Motion Comparator is built as a convenient tool for Robot Operating System (ROS) users. Motion Comparator can directly parse binary `rosbag` files that record robot motions. In addition, Motion Comparator also reads motions from `csv` files, allowing users to visualize motions exported from other platforms.

*3) Customizable layouts:* As shown in Fig. 1, Motion Comparator is multi-view application with several panels. The layout is customizable, *i.e.*, allowing users to create, reposition, stack, or destroy views. The customizable layouts allow users to configure the tool to best address their needs, such as creating multiple coordinated views for visualization and various juxtaposition designs for comparison.

*4) Save, restore, and share:* In Motion Comparator, a workspace contains all changes made by users, including loaded motion data, created visualizations, and customized layouts. A workspace can be saved to a file and shared by a link. By clicking on the link, a roboticist can restore the exact same workspace, seeing the same robot motion data in the same visualizations using the same layout, thereby facilitating the communication of robot motions.

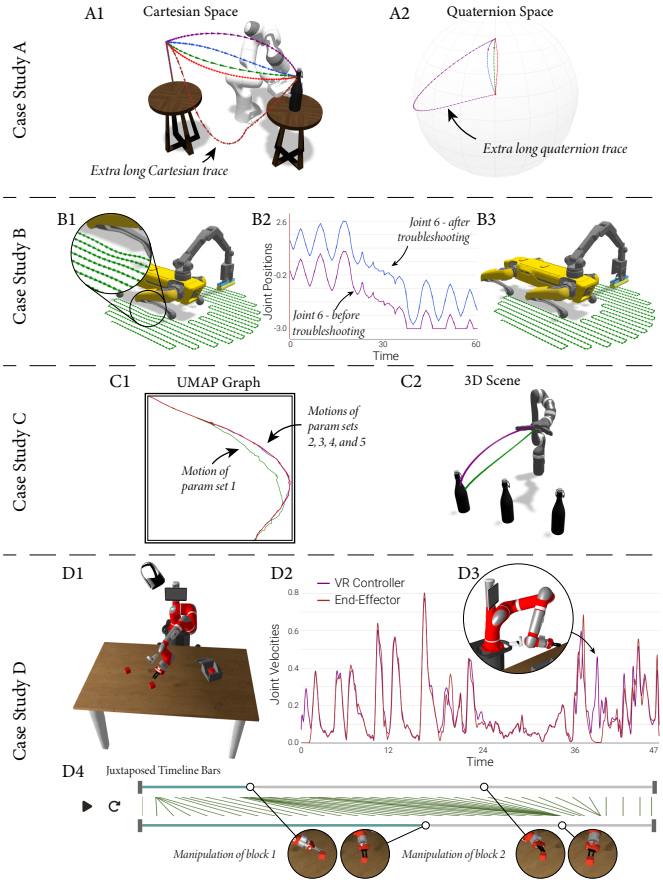


Fig. 3. Visualizations for case studies in §VII.

## VII. Validation

Motion Comaraptor is designed for roboticists to compare robot motions. For this kind of visualization systems that support solving real-world problems faced by domain experts, the most common form of validation are case studies with real users, real problems, and real data [31]. In this section, we present four case studies in which Motion Comparator is utilized for motion selection, troubleshooting, parameter tuning, or motion review. Additionally, we invited two roboticists to use Motion Comparator to perform the tasks in the case studies. Both roboticists flexibly used features in our tool, including position traces, quaternion traces, UMAP graphs, and juxtaposition designs to compare motions. Their insights and feedback were collected and incorporated into our system to enhance its usability. In addition, we also used our tool as an educational tool to explain basic concepts of robot motions to a novice robotics student. All the case studies can be interactively explored using our online demo page[3].

### A. Motion Selection after Planning

Many common planning algorithms involve randomization. It is a widely used approach to generate multiple motion candidates using one or more motion planners, and subsequently select one motion to deploy to the physical robot.

[3]Demo page: https://pages.graphics.cs.wisc.edu/MotionComparator

Suppose a roboticist uses a sampling-based motion planner, Transition-based Rapidly-exploring Random Trees [32], to generate motions for a Franka Emika Panda robot to pick up a water bottle and place it elsewhere. The randomness of the sampling-based motion planner means that it generates a different motion at each run. Without Motion Comparator, the standard approach typically requires the roboticist to play each motion one by one in a visualization tool such as Rviz. The roboticist needs to memorize all the motions and mentally compare them. Using Motion Comparator, the position traces first exclude one motion because it involves redundant movements (Figure 3-A1). Afterward, another motion is disregarded because of its significantly longer quaternion trace (Figure 3-A2). Longer quaternion traces indicate more rotation is applied to the bottle. Finally, the 3D scenes of Motion Comparator enable the roboticist to examine the remaining three motions and identify the best motion to deploy.

### B. Troubleshooting

Despite the existence of sophisticated motion planning algorithms, roboticists still play a crucial role in determining many aspects of motion generation, including starting configurations, placements of a robot, and ways to attach end-effectors. These decisions directly impact the qualities of robot motions. Motion Comparator helps roboticists troubleshoot and improve motion qualities through various visualizations. In this case study, a roboticist programs a Boston Dynamics Spot robot equipped with a robot arm for a floor-wiping task. After parking at certain locations, the robot wipes nearby areas using the cleaning tool in hand. The arm follows a pre-defined lawn-mower path using an optimization-based inverse kinematics solver, *RangedIK* [33]. There are multiple ways to attach the cleaning tool to the robot's hand and the roboticist chooses one according to their experience. The position trace in Motion Comparator indicates that the robot is not capable of constantly following the given path (Figure 3-B1). Without Motion Comparator, the roboticist may manually develop code to inspect joint positions, velocities, accelerations, or jerks. With multiple visualization features being integrated into Motion Comparator, the roboticist can identify the problem in the 3D scene, navigate to the problematic segment using the timeline bar, and troubleshoot using a time-series plot. The time-series plot reveals that joint 6 of the robot's arm is restricted by its lower position limit (Figure 3-B2). Knowing the cause, the roboticist chooses an alternative way to mount the tool. The robot is able to accurately track the sweeping path now that joint 6 is no longer limiting it (Figure 3-B3).

### C. Parameter Tuning for Legible Motions

Many motion generation methods involve parameters that need manual tuning, *e.g.*, the weights in a weighted-sum multiple-objective optimization-based approach [34], the hyperparameters in reinforcement learning-based approaches [35], and the heuristic parameters in sampling-based motion planners [36]. In order to understand the ramifications of parameters, roboticists need to *compare* the motions generated using different parameters. A roboticist utilizes an optimization-based motion generator [34] to generate legible motions to pick up a bottle. Five sets of parameters generate five motions. Without Motion Comparator, the roboticist may initiate five instances of visualization tools (*e.g.*, Rviz) to display the five motions side-by-side. The roboticist needs to manually synchronize camera viewpoints and temporally align the motions. Using Motion Comparator, the UMAP graph view reveals that the motions generated by four parameter sets are in close proximity (Figure 3-C1). As opposed to comparing all five motions, the roboticist simply compares two of them. Subsequently, the roboticist visually determines the legibility of motions using position traces (Figure 3-C2). Our tool enables the roboticist to identify the set of parameters that can produce the most legible motion.

### D. Review of Motions in Teleoperation

Motions in the aforementioned case studies are generated by planning algorithms. Robot motions can also be generated through programming by demonstration approaches such as teleoperation [37]. In this case study, a roboticist develops a teleoperation system in which an operator utilizes a VR controller to teleoperate a Rethink Robotics Sawyer robot to pick up blocks and drop them in a box (Figure 3-D1). In order to inspect the system's performance, the roboticist conducts a comparative analysis between input human motions and output robot motions. The time-series plot in Motion Comparator shows that the system has consistently small latencies (Figure 3-D2). The inconsistency in the plot helps the roboticist detect when the robot loses the capability to accurately mimic its operator due to singularities or self-collision (Figure 3-D3). In order to gain insights into different operation strategies, *e.g.*, the ways to pick up a block, the roboticist also needs to compare motions generated by various users. Without Motion Comparator, the roboticist may need to manually navigate through motions to observe how the robot picks up a block. Using Motion Comparator, the time warping feature offered by Motion Comparator enables the roboticist to quickly align two motions and compare various ways to pick up a block (Figure 3-D4).

## VIII. Discussion and Conclusion

In this section, we discuss the limitations of our work and provide a conclusion of the paper. Our work has a number of limitations that highlight potential directions for future research. First, while our system supports visual comparison among multiple robot motions, its usability significantly declines with a large number of motions. Neither juxtaposition nor superposition designs scale to many objects [1]. In addition, our time warping and explicit encoding designs are limited to two-way comparisons. Future work should explore the approaches to visualize and compare numerous robot motions. Second, Motion Comparator utilizes Uniform Manifold Approximation and Projection (UMAP) to reduce the high dimensionality of joint traces. Being a general dimension reduction tool, UMAP is not specifically designed for robot

joint data and may produce discontinuous traces despite the input joint traces being continuous. Future research can explore novel dimension reduction approaches to reduce the dimensionality of robot joint data. Third, while providing several visualization approaches, Motion Comparator places the burden on the user to select, combine and apply them to their problem. Future work should focus on the usability, learnability, and effectiveness of the approaches in supporting motion comparison.

Motion Comparator is a visualization tool that benefits roboticists by facilitating comprehension, comparison, and communication of robot motions. Our design process involves a literature-based requirement analysis, the adaptation of multiple visualization approaches for motion comparisons, and the application of multi-view coordination approaches to integrate these visualizations. We believe that our tool and the design ideas it embodies contribute to robotic research by directly facilitating motion comparison and informing the development of future robot visualization tools.

## REFERENCES


[1] M. Gleicher, "Considerations for visualizing comparison," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 413–423, 2017.

[2] Z. Kingston and L. E. Kavraki, "Robowflex: Robot motion planning with moveit made easy," in *2022 IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 3108–3114.

[3] D. Coffey, F. Korsakov, M. Ewert, H. Hagh-Shenas, L. Thorson, A. Ellingson, D. Nuckley, and D. F. Keefe, "Visualizing motion data in virtual reality: Understanding the roles of animation, interaction, and static presentation," in *Computer Graphics Forum*, vol. 31, no. 3pt3. Wiley Online Library, 2012, pp. 1215–1224.

[4] L. Reveret, S. Chapelle, F. Quaine, and P. Legreneur, "3d visualization of body motion in speed climbing," *Frontiers in Psychology*, vol. 11, p. 2188, 2020.

[5] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002, pp. 473–482.

[6] C. Ware, R. Arsenault, M. Plumlee, and D. Wiley, "Visualizing the underwater behavior of humpback whales," *IEEE Computer Graphics and Applications*, vol. 26, no. 4, pp. 14–18, 2006.

[7] D. Schroeder, T. Kowalewski, L. White, J. Carlis, E. Santos, R. Sweet, T. S. Lendvay, T. Reihsen, and D. F. Keefe, "Exploratory visualization of surgical training databases for improving skill acquisition," *IEEE Computer Graphics and Applications*, vol. 32, no. 6, pp. 71–81, 2012.

[8] B. Russig, D. Graß, R. Dachselt, and S. Gumhold, "On-tube attribute visualization for multivariate trajectory data," *IEEE Trans. on Visualization and Computer Graphics*, vol. 29, no. 1, pp. 1288–1298, 2022.

[9] A. J. Hanson and H. Ma, "Visualizing flow with quaternion frames," in *Proceedings Visualization'94*. IEEE, 1994, pp. 108–115.

[10] B. Jüttler, "Visualization of moving objects using dual quaternion curves," *Computers & Graphics*, vol. 18, no. 3, pp. 315–326, 1994.

[11] V. V. Patel, D. Rakita, and A. M. Dollar, "An analysis of unified manipulation with robot arms and dexterous hands via optimization-based motion synthesis," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 8090–8096.

[12] A. J. Hanson, *Visualizing Quaternions*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

[13] D. Rakita, B. Mutlu, and M. Gleicher, "Motion synopsis for robot arm trajectories," in *2016 IEEE International Sympo. on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2016, pp. 281–287.

[14] S. Johansen, J. Donovan, and M. Rittenbruch, "Illustrating robot movements," in *HRI'23: Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*. Association for Computing Machinery (ACM), 2023, pp. 231–242.

[15] 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.

[16] K. Kim, J. V. Carlis, and D. F. Keefe, "Comparison techniques utilized in spatial 3d and 4d data visualizations: A survey and future directions," *Computers & Graphics*, vol. 67, pp. 138–147, 2017.

[17] N. Vaughan and B. Gabrys, "Comparing and combining time series trajectories using dynamic time warping," *Procedia Computer Science*, vol. 96, pp. 465–474, 2016.

[18] L. Gong, B. Chen, W. Xu, C. Liu, X. Li, Z. Zhao, and L. Zhao, "Motion similarity evaluation between human and a tri-co robot during real-time imitation with a trajectory dynamic time warping model," *Sensors*, vol. 22, no. 5, p. 1968, 2022.

[19] T. Giorgino, "Computing and visualizing dynamic time warping alignments in r: the dtw package," *Journal of Statistical Software*, vol. 31, pp. 1–24, 2009.

[20] A. Gogolou, T. Tsandilas, T. Palpanas, and A. Bezerianos, "Comparing similarity perception in time series visualizations," *IEEE trans. visualization and computer graphics*, vol. 25, no. 1, pp. 523–533, 2018.

[21] W. Gomes, V. Radhakrishnan, L. Penco, V. Modugno, J.-B. Mouret, and S. Ivaldi, "Humanoid whole-body movement optimization from retargeted human motions," in *2019 IEEE-RAS 19th International Conf. on Humanoid Robots (Humanoids)*. IEEE, 2019, pp. 178–185.

[22] T. P. Kucner, M. Magnusson, E. Schaffernicht, V. H. Bennetts, and A. J. Lilienthal, "Enabling flow awareness for mobile robots in partially observable environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1093–1100, 2017.

[23] K. Guruprasad and D. Ghose, "Multi-agent search strategy based on centroidal voronoi configuration," in *2010 IEEE International Conf. on Robotics and Automation*. IEEE, 2010, pp. 3550–3555.

[24] H. Zhang, H. Chen, N. Xi, G. Zhang, and J. He, "On-line path generation for robotic deburring of cast aluminum wheels," in *2006 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2006, pp. 2400–2405.

[25] M. Phillips, A. Dornbush, S. Chitta, and M. Likhachev, "Anytime incremental planning with e-graphs," in *2013 IEEE International Conf. on Robotics and Automation*. IEEE, 2013, pp. 2444–2451.

[26] L. Palmieri, R. Andrey, J. Mainprice, M. Hanheide, A. Alahi, A. Lilienthal, and K. O. Arras, "Guest editorial: Introduction to the special issue on long-term human motion prediction," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5613–5617, 2021.

[27] M. J. Gielniak and A. L. Thomaz, "Spatiotemporal correspondence as a metric for human-like robot motion," in *Proceedings of the 6th international conf. on Human-robot interaction*, 2011, pp. 77–84.

[28] P. Senin, "Dynamic time warping algorithm review," *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, vol. 855, no. 1-23, p. 40, 2008.

[29] S. Milanko, "Wuda: Visualizing and transforming rotations in real-time with quaternions and smart devices," in *2023 IEEE Visualization and Visual Analytics (VIS)*. IEEE, 2023, pp. 241–245.

[30] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[31] M. Sedlmair, M. Meyer, and T. Munzner, "Design study methodology: Reflections from the trenches and the stacks," *IEEE trans. visualization and computer graphics*, vol. 18, no. 12, pp. 2431–2440, 2012.

[32] D. Devaurs, T. Siméon, and J. Cortés, "Enhancing the transition-based rrt to deal with complex cost spaces," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 4120–4125.

[33] Y. Wang, P. Praveena, D. Rakita, and M. Gleicher, "Rangedik: An optimization-based robot motion generation method for ranged-goal tasks," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 8090–8096.

[34] C. Bodden, D. Rakita, B. Mutlu, and M. Gleicher, "A flexible optimization-based method for synthesizing intent-expressive robot arm motion," *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1376–1394, 2018.

[35] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[36] S. R. Lindemann and S. M. LaValle, "Current issues in sampling-based motion planning," in *Robotics Research. The Eleventh International Symposium: With 303 Figures*. Springer, 2005, pp. 36–54.

[37] Y. Wang, C. Sifferman, and M. Gleicher, "Exploiting task tolerances in mimicry-based telemanipulation," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 7012–7019.