Animation Key-framing using Alternate Reality

COBEY HOLLIER* and GLEN MCMANUS*, University of Victoria, Canada

Additional Key Words and Phrases: alternate reality, animation, key-framing, CST

ACM Reference Format:
Cobey Hollier and Glen McManus. 2022. Animation Key-framing using Alternate Reality. *ACM Trans. Graph.* 37, 4, Article 111 (August 2022), 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In recent years video games and digital animation have skyrocketed in popularity. A study in 2020 found that the video game market is expected to grow by 13% a year until at least 2027. [5] Due to this growth, an increasing number of people are getting involved in the gaming industry meaning that many new people are learning game development tools such as the Unreal Engine and Unity. These tools require a steep learning curve and are often unapproachable to people looking to dabble in making their first game. Furthermore, creating prototypes collaboratively is challenging because a computer only provides a single mouse to interact with. This project aims to provide a collaborative and engaging way to create game scenes and animations without requiring the user to have extensive game development experience. Our primary target is junior software developers who have little-to-no experience with Unity and non-technical managers at game development studios. In the future this could be extended into 3D movie animation studios and other 3D scene applications.

1.1 Design Principles

This project was guided by several Creativity Design Tool principles that shaped the form of the final product. The primary design principles that were incorporated include the following:

(1) Support exploration, finding inspiration and answers. Game development is an inherently creative endeavour which lends itself to tool that encourage exploration to find creativity.
(2) Low threshold, making game development more approachable for the general population.
(3) Supporting collaboration. The main challenge identified in collaboration is the limited input that a mouse provides.
(4) Wide walls. The product was designed to give beginners an entry point and allow experts to perform their job more easily. Even an expert needs time to place objects in orient them correctly in a game scene.

These design principals will be discussed in greater detail later in the report as the implementation is described.

2 RELATED WORK

This project drew on many related works that guided the design decisions. The first work discusses the difficultly of building 3D Scenes using conventional computer input. [2] They state that experts spend significant time learning 3D modelling tools for movies, video games and interior design. In this study they consider an alternative approach using natural language to input objects into a scene. They found promising results that alternative forms of input could be used to generate scenes. In their conclusion they state that their input provides a quick and easy starting point that could then be later refined with more sophisticated tools.

Our second study compares the aesthetics, usability, engagement, social connectivity and enjoyment of physical vs digital tabletop games. [1] Tabletop games provide a strong analogy to the the process of 3D scene creation because both require laying out physical information in an environment. The authors of the paper found that digital mediums still cannot compare to their physical counterparts when comparing social connectivity and engagement. They found that digital versions required use of third party programs such as Discord to shoulder the brunt of the communication. This provided a worse experience overall than interacting with physical objects with other users. This is an encouraging result in the context of building a physically interactive method for collaboratively building game scenes.

3 DESIGN FICTION

3.1 Chapter 1: Autonomous Markers

When retouching or iterating scenes, how can a user edit frames without modifying each keyed value? How can a user quickly place markers in a scene according to where their corresponding GameObjects are positioned?

Jo comes back from lunch to iterate a cinematic sequence they blocked out earlier in the day with the product owner, Pat, of the unannounced title their company is producing. After getting started, an emergency meeting is called with the product owner. There's been a rewrite and half of a previously completed scene needs to be redone. Jo and Pat load up the scene, and scrub through the timeline with the ultrasonic. The ArUco markers slide and rotate around the workspace as the timeline is scrubbed, coming to rest in final 'good' position in the sequence. They re-zero the timeline to constrain the scope of their timeline, so that they can't scrub back in time before that frame. As they iterate the sequence the markers move and rotate with the keyed values in the timeline. They can discuss

*Both authors contributed equally to this research.

Authors' address: Cobey Hollier, cobeyhollier@uvic.ca; Glen McManus, gmcmanus@uvic.ca, University of Victoria, V8P 5C2, Victoria, British Columbia, Canada, 43017-6221.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2022 Association for Computing Machinery.
0730-0301/2022/8-ART111 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

the qualities that were working before and preserve those which still work in the newest version of the sequence.

3.2 Chapter 2: Gestures

For a skilled user of the AR animation key-framing utility, how can the workflow accommodate advanced usage and present more attractive and efficient usage?

Francais is iterating an in-game cinematic. Blocked out action was delivered by the game design team, and now Francais will polish the sequence. Additionally, a storyboard containing concept art is delivered to Francais to direct the visual qualities of the sequence (such as lighting, and VFX). Francais scrubs through the blocked out timeline with the ultrasonic; the markers used in the scene position and rotate themselves accordingly as the frames update. Francais does a pinching gesture over the marker to reveal a context menu related to the marker's object. Now the object will play a predefined animation at frame x, and transition into a new predefined animation at frame y while moving along the blocked out path. Another gesture targets the global illumination settings of the scene, and Francais can adjust the direction of light by pointing a finger in the direction the light should travel. Making the universal colour picking gesture, Francais can choose the colour of the directional light, and ambient light; Francais immediately sees how these changes effect the scene.

4 PROTOTYPE DESIGN AND IMPLEMENTATION

The implementation of this project involves a combination of hardware, software and electrical elements that are integrated together into a final product. The project aims to create a playground for creating 2D animations using key-frames. Key framing is an animation technique that reduces the amount of labor required for each frame of the video. Instead of creating a separate drawing or scene for every frame in the animation a "Key Frame" is created at the important moments in the animation and software interpolates between the frames. For example, if somebody was trying to animate a person standing up they would key-frame the person sitting on a chair, the person half way standing and the person fully standing. The software could then generate the intermediary frames using interpolation.

While key-framing reduces the amount of work by reducing the number of frames that need to be modelled, it doesn't reduce the difficultly of creating the important frames.

4.1 ArUco Markers

To reduce the complexity of creating a scene for each frame, a collection of Fiducial markers called ArUco markers are used. Fiducials are symbols that are easily recognizable to a computer vision camera. More specifically, ArUco markers are binary square Fiducial markers that allow camera pose estimation. OpenCV's ArUco library provides a method for generating ArUco markers given the grid size (dictionary name), marker side length (pixels), border bits (pixels), and (integer) id. [4] Figure 1 shows a screenshot of the creation components, with attention to the Dictionary Name, and Marker Id which map the marker to the ID and $N \times N : 4 <= N <= 7$ grid to which the ID belongs.

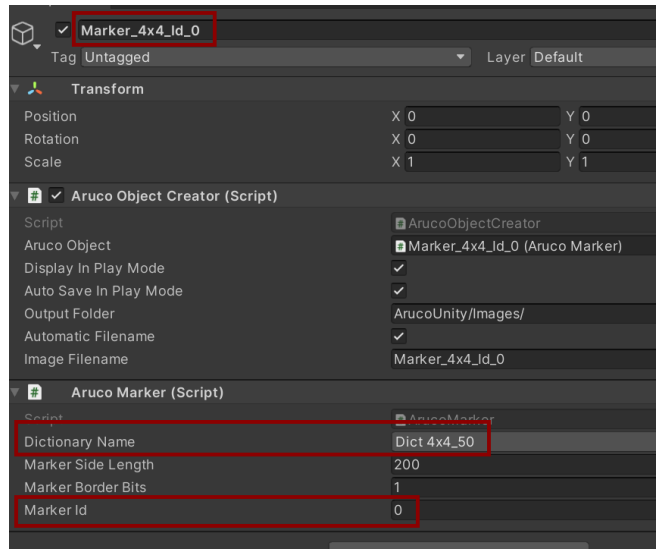

Fig. 1. Marker Creation Component

Fig. 2. ArUco Marker (Dict4x4_50, ID 0)

With the above settings, an ArUco marker similar to the one in Figure 2 is output. The ArucoUnity wrapper provides a Unity scene to prepare the markers. The markers are printed onto square tiles around 3cm across. The user can place these Fiducials on a 2D plane in any orientation and translation they want. Each of these Fiducials will later be mapped to an object in the virtual game scene on the computer.

4.2 AR Camera

A smartphone camera is then used to scan the Fiducials into Unity using the package ArUco Unity [3] created by Normand. To begin, a ChArUco board (ArUco board with chessboard corners, see Figure 3) is used for calibration of the camera.

Calibration is done once per camera to help the image processing pipeline deal with distortions of the input image by "measur[ing] the camera's parameters." [3] Once calibrated, the markers can be tracked. The video feed from the camera is processed with OpenCV through the C# wrapper created by Normand. The position of the markers is provided by OpenCV's "detectMarkers," and the rotation of markers is estimated by "estimatePoseSingleMarkers." [4]

4.3 Unity Animation

Each of the ArUco markers is mapped to a Unity GameObject. These GameObjects have ArucoMarker components as shown in Figure 1.

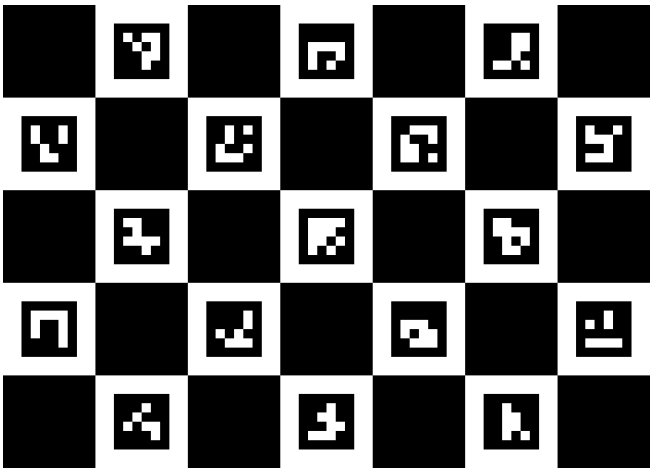

Fig. 3. Example ChArUcoBoard (Dict4x4, 50px, x7 y5)

To map these ArucoMarker components and respective GameObjects to the timeline, we have created the ArucoToTimeline component (Figure 4).

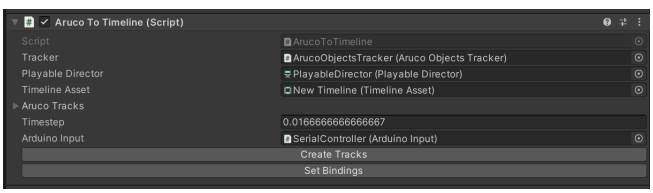

Fig. 4. Aruco To Timeline Component

Given an ArucoObjectsTracker (provided by the ArucoUnity package), and a Unity PlayableDirector component, we generate tracks ("Create Tracks" button) for the timeline bound in the PlayableDirector component. These tracks get bound ("Set Bindings" button) to the corresponding ArucoMarker components. The resulting timeline will look similar to Figure 5, where we have the name and id of the markers identifying the tracks and clips of the timeline.

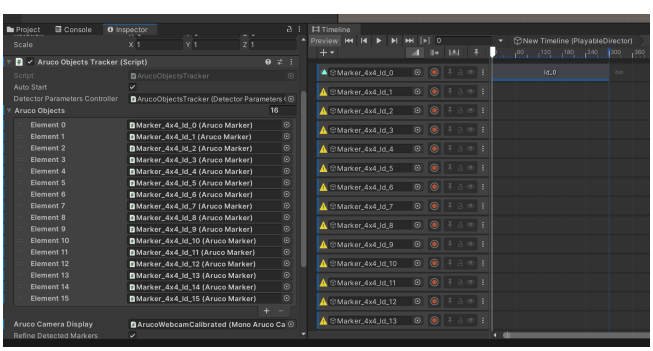

Fig. 5. Aruco objects mapped to timeline

While detecting the ArUco markers, the user may press the 'K' key to insert a keyframe. The keyframe stores position data for the x and z axis, and rotation data along the y axis. This allows us to block out the movement of objects in a scene and explore different flows of motion for agents over time in a scene. The data generated through the blocking process directly integrates with the standard Unity Animation pipeline (Figure 6), hence, it may be retouched and extended with complex behaviour as needed to suit the needs of the project in production. [7]

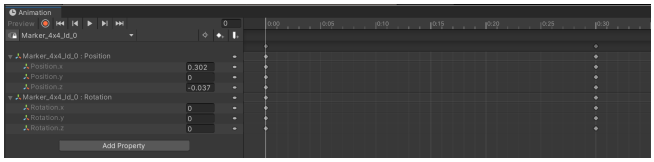

Fig. 6. Marker Animation Clip

4.4 Timeline scrubbing

When creating a key-frame based animation the creator must scrub through the timeline to see the frames they have already created and to choose the location of their next frame. Not every frame will be the same distance apart because high-action scenes will have more tightly packed frames than scenes where little is changing. To solve this problem, a physical timeline scrubber was created that allows the users to collaboratively navigate the timeline.

The timeline scrubber relies heavily on the Ultrasonic sensor. The ultrasonic was mounted on one end of a 60cm long track built out of thin Medium Density Fibreboard (MDF) as seen in the figure below.

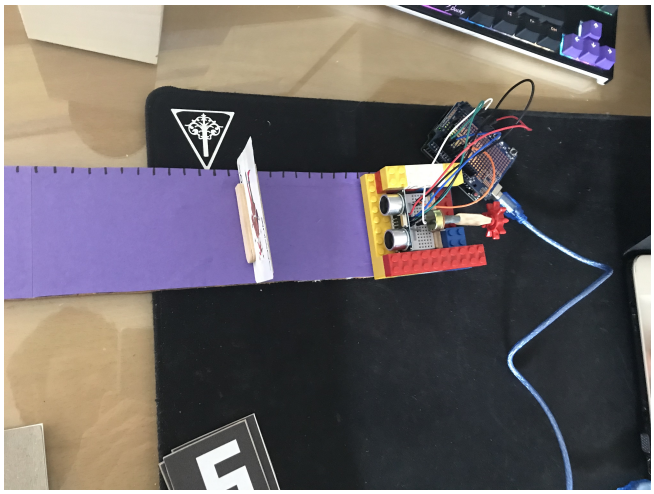

Fig. 7. Timeline Scrubbing

This track is a physical representation of the virtual timeline in Unity. By moving a vertical barrier along the track, the digital timeline scrubs to the corresponding position in Unity. This allows the animators to choose the position of their next key-frame or overwrite a frame they have already recorded. The advantage of a physical solution over a pure software solution is the affordance to collaboration and communication. A pure software solution requires

the users to share a mouse to input their information. Additionally, the timeline in Unity isn't very intuitive to use.

4.5 Sensitivity Adjustment

The physical slider is an improvement over a software solution, but its precision is limited by the accuracy of the ultrasonic sensor and the physical ability of the user. To improve this system, a potentiometer was added that could adjust the scale of the timeline. This alters the number of frames that are skipped per 1cm of physical track. If you need many key-frames then a high precision can be used, whereas an animation with only a few key-frames can use a low precision.

4.6 Implementing Design Principles

This section discusses how this project implemented the design principles discussed in section 1.1.

4.6.1 Exploration. Exploration is a fundamental principle of our 3D modelling system. The simplicity of manipulatives and ease of creation allows the users to explore without wasting vast amount of time. Furthermore, since the fiducials can be mapped to any digital asset, they can be re-purposed for other projects or idea. This principal of reuse lowers the cost of experimentation which encourages the user to explore the environment freely.

4.6.2 Collaboration. Collaboration is the main motivation of this project that governed most of the design decisions. Layout tiles allow for easier collaboration by providing the users with better concurrent input. Each user can grab a different Fiducial tile and arrange them at the same time.

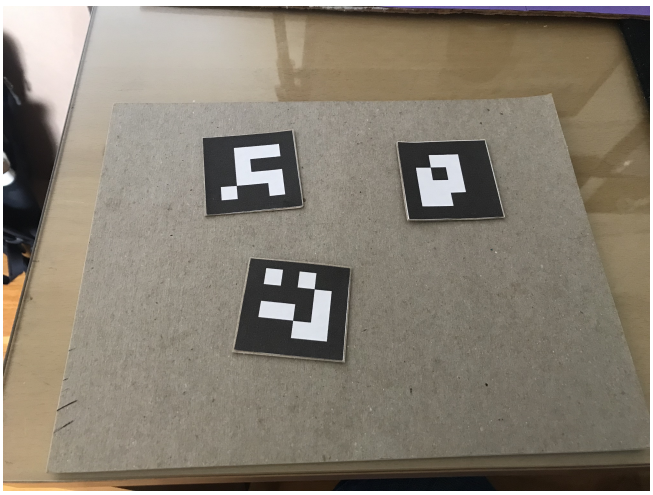

Fig. 8. Multi-Markers

The speed of animation layout also allows users to quickly prototype and demonstrate their ideas to their peers. The physical key-frame slider also allows the users to easily see and manipulate the current frame without trying to all control the same computer. A physical device can be better for this kind of task because it can be viewed from all angles (unlike a computer screen) which allows users to sit around a table together and all partake in the discussion.

4.6.3 Low Threshold. This project lowers the threshold to animate 3D scenes by eliminating the learning curve of animation software. Everybody learns how to manipulate objects in physical space at a young age because that is part of our daily lives. This project makes the animation processes closer to experiences we experience in our regular lives. This principle is important to our project because we envision developers and management working together to mock-up scenes. The developers don't have the authority to make decisions so lowering the threshold to the level of the designers can allow more people to get actively involved.

5 CRITIQUE

This section will discuss the relative strengths of weaknesses of our project implementation. Every project has limitations and is useful within a specific scope.

5.1 Strengths

The main strengths of this project are its ease-of-use and flexibility. The modular nature of the fiducials allow them to be mapped to any digital object. Moreover, they afford the creation of very quick animations that can be done collaboratively. This allows users to easily share ideas. Additionally, the integration of the ultrasonic sensor to scrub the timeline provides a convenient way to review the work done so far. The user can see the animation update forward and in reverse. Furthermore, there is no need to target the small timeline marker to change the current frame. All members exploring ideas together who are in arms reach of the ultrasonic can quickly put their hand in to point to the frames they want to review with their team.

5.2 Limitations

While the project has many advantages, it has limitations introduced by time and resource constraints. The most obvious weakness is the precision of the object tracking. The tracking code makes use of an external library that causes the objects to vibrate in the scene even when the camera and objects are stationary in the physical world. Moreover, this library only supports Windows which prevents the project from being used on a Macintosh computer without modifications. Additionally, sometimes there is an issue with the unmanaged native code which crashes Unity.

The scanner is also limited to 2 dimensions of translation and 1 dimension of rotation. The scene generates in 3 dimensions, but all the objects are placed on a 2D grid. The current implementation doesn't provide a way to elevate objects above each other because that would block line-of-sight to the camera.

6 FUTURE WORK

This project could be further improved by stabilizing the OpenCV wrapper, and including binaries for platforms beyond Windows, such as Linux and Mac. As mentioned in limitations, input for additional dimensions is missing; support for a second camera could be added, along with a change to using cube ArUco markers would

allow for manipulation of objects in 3 dimensions. This would accommodate finer grain control over objects in a 3d space where all position and rotation elements can be keyed concurrently. More of the OpenCV library could be leveraged to accommodate more functionality, providing a "higher ceiling" for the skill cap of this tool. As mentioned in the design fiction, hand gestures would be a good addition. Hand gestures would allow quick and intuitive access to the properties of the objects which are mapped to the ArUco markers. [6]

In Unity, animations key frames can be controlled with curves. This is helpful for non-linear interpolation of keyed values. It would be really nice to be able to draw a curve on a whiteboard, and apply that to the key frames. This would be helpful after the initial blocking of positions is done, during some of the early work in polishing the action. Manipulating the key frame curves in the GUI can be a bit tedious, as it requires adjusting tangents to get the right curve. If instead, one could quickly draw a curve it would provide a much lower threshold, and exploratory mode for prototyping animations.

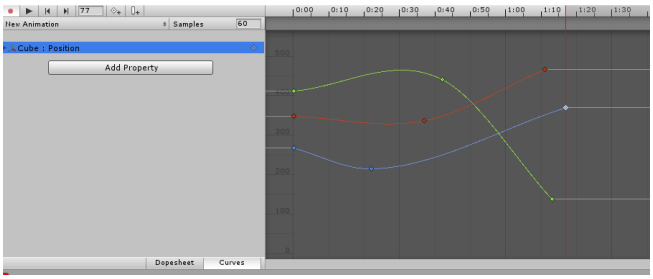

Fig. 9. Unity Animation Curve

Another extension could be added to transfer more of the model creation into physical manipulative instead of relying on software. For example, it would be interesting if the user could cut out rectangles, triangles and circles that could be scanned directly into Unity without the use of Fiducials. This would give the user more visual context to the relative position and rotation of the objects in the scene without looking at the computer monitor. This would also allow users to cut complicated shapes out of paper and scan them into Unity as models instead of relying on their 3D modelling skills.

In the future additional features could be added to tailor to different users. For example, we could develop features tailored towards interiors designers who are trying to quickly build layouts for a house or architects trying to quickly put together room layouts.

REFERENCES

[1] Jonas Ekblad Andreas Larsson. 2020. *Physical vs Digital Tabletop Games.* Retrieved March 26, 2022 from https://www.diva-portal.org/smash/get/diva2:1480675/FULLTEXT01.pdf

[2] Manolis Savva Angel X. Chang, Mihail Eric and Christopher D. Manning. 2017. *SceneSeer: 3D Scene Design with Natural Language.* Retrieved March 26, 2022 from https://arxiv.org/abs/1703.00050

[3] Erwan Normand. 2019. *Normanderwan/Arucounity: Bring Augmented Reality to unity by tracking aruco markers in real time.* Retrieved March 26, 2022 from https://github.com/NormandErwan/ArucoUnity

[4] OpenCV. n.d. *Detection of ARUCO markers.* Retrieved March 31, 2022 from https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html

[5] Grand View Research. 2020. *Video Game Market Size, Share & Trends Analysis Report By Device (Console, Mobile, Computer), By Type (Online, Offline), By Region, And Segment Forecasts, 2020 - 2027.* Retrieved March 24, 2022 from https://www.grandviewresearch.com/industry-analysis/video-game-market#:~:text=The%20global%20video%20game%20market,key%20factors%20driving%20the%20growth

[6] Murtaza's Workshop Robotics and AI. 2021. *Multiple hand gesture control with opencv python.* Retrieved April 1, 2022 from https://www.youtube.com/watch?v=3xfOa4yeOb0

[7] Unity Technologies. n.d. *Animation clips.* Retrieved March 31, 2022 from https://docs.unity3d.com/Manual/AnimationClips.html