StoryboardXR Project Report

Mixed reality shot planning and preproduction tool

KENNETH J. YANG, DALTON BROCKETT, and MARLEY BYERS, University of Washington

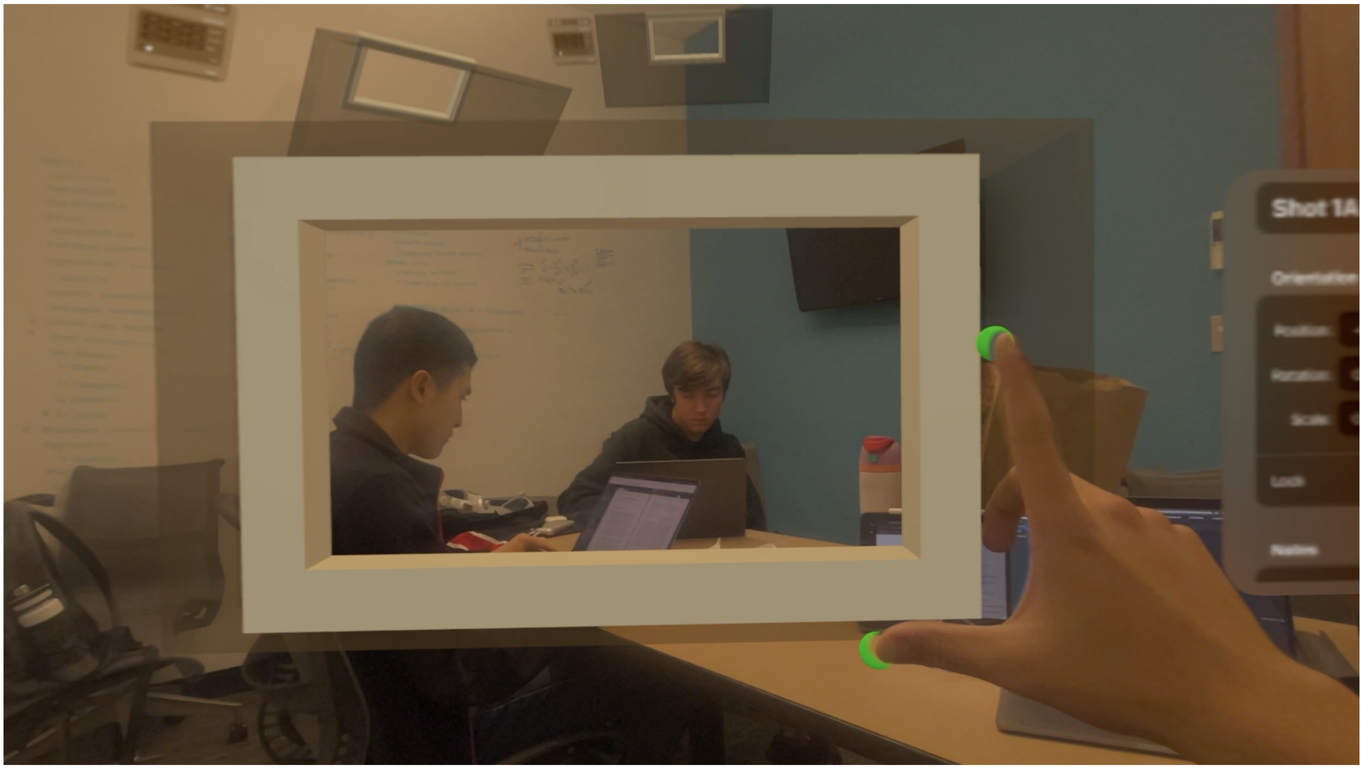


Fig. 1. A user of StoryboardXR adds a frame to their view using the hand gesture, planning for a shot in the scene.

StoryboardXR is a mixed reality application designed to enhance the filmmaking preproduction process by enabling filmmakers to plan their shots in 3D on location. Traditional storyboarding requires encoding 3D spatial information on a 2D drawing surface. This is intuitive and we believe mixed reality can be used to improve this. Using intuitive hand gestures, users can spawn 3D view frusta to plan their scene layout in the natural environment of their set. Shots will be tracked relative to a world anchor and scenes can be saved, reloaded, and shared. StoryboardXR is written in Swift for the Apple Vision Pro.

1 INTRODUCTION

We propose a mixed-reality application for filmmakers to plan and test camera shots on a film set. Our application aims to enable filmmakers to plan their shoots in the native 3D environment they will be working in. While on set, users wearing an Apple Vision Pro can place virtual shot frames around them and see spatially what their coverage of a scene is like.

Filmmakers use storyboards to plan out shot sequences, camera angles, transitions, action direction, and more. Storyboards, however, must rely on extra notation to encode these inherently 3D plans onto a 2D drawing environment. When drawing in a storyboard as well, a new reality is created that may not accurately reflect what is available on set or at location. Our application aims to provide the planning features needed by filmmakers in their native 3D context and enable more consistent plan-to-shot experiences and better preproduction visualization tools.

We will target the Apple Vision Pro to build this application. The Apple Vision Pro provides high quality video pass-through and excellent 3D rendering and immersion, required for planning films. The Apple Vision Pro also provides solid world tracking, crucial for accurate shot planning and consistency when sharing plans with others.

StoryboardXR provides filmmakers with novel tools to plan their shots and enables a new paradigm for accelerating the preproduction process.

1.1 Contributions

- Dalton: Custom gesture recognition for frame spawning
- Dalton: Hand tracking system

Authors' address: Kenneth J. Yang, kjy5@uw.edu; Dalton Brockett, daltonbo@uw.edu; Marley Byers, m4rl10@uw.edu, University of Washington.

- Kenneth: Place shot frames in 3D space, with world space anchoring.
- Kenneth: Save and reload scenes, and export them as JSON files.
- Kenneth: Application navigation and user interfaces attached to frames.
- Marley: Add 3D models for blocking.
- Marley: Convert json file of scene to 2D visualization of shots from topdown view.

2 RELATED WORK

Currently in industry there are many of previsualization software tools, including some designed for VR. These tools however do not address creating framings and transitions on-set and rather rely on virtual proxy sets. The closest tools are Tvori [Tvori 2025] and FirstStage [Moviestorm 2024] which provide VR-based previsualization and shot planning. They both however, lack the mixed-reality aspect of planning with the same shooting set. Conventional applications such as StudioBinder [StudioBinder 2025] focus on overall content management in the filmmaking process and provides digital alternatives to paper-based storyboarding and related workflows. While these help streamline the process, but again do not enable planning using the actual geometry of the set. A related academic project targeting previsualization is CollageVis [Jo et al. 2024]. This project focuses on building previsualizations through piecing together videos and arranging them in a virtual stage. While there are projects and products surrounding the field of previsualization and film planning, there does not appear to be a product that achieves our goals of planning directly in the set of the shoot.

Spenser Sakurai on Youtube has a video [Sakurai 2022] on storyboarding in VR using Wonder Unit's Storyboarder. While his method implements a lot of the features we have or (or would add with more time), it requires that you first capture the environment with something like Polycam and then you can storyboard in the captured virtual environment. Our implementation allows you to storyboard live without creating the environment in 3D beforehand.

3 METHOD

3.1 Shot Planning

In StoryboardXR, users open and modify scenes. Scenes are comprised of various shots. To plan a shot, users add a shot frame, depicted as a 3D frusta into the scene. Frames spawn in with the same orientation as the user's head letting planners plan their shots exactly from their perspective. Through simple gestures, users can manipulate the position, orientation, and scale of frames to plan out their scene.

Now that they are situated in a 3D environment, users can also step back and see the coverage they get with their plan. Using the user interface, users can save their scenes to their device and reload them later. Once a scene is loaded it can also be shared as a JSON file to other programs for processing.

3.2 Handtracking

In order for frame spawning to seamlessly integrate into the workflow of shot planning, we leverage the different joints of our tracked hands to register a specific custom gesture: an L shape with one hand and tap with the other. For the L gesture specifically, we map vectors to the index finger and thumb of a given hand and check to see if the dot product is "close" to being orthogonal(i.e. the angle between the vectors is within some degree threshold).

$$\vec{v}_{\mathrm{I}} = \vec{v}_{\mathrm{index}} = \vec{p}_{\mathrm{tip}}^{\mathrm{index}} - \vec{p}_{\mathrm{metacarpal}}^{\mathrm{index}}$$

$$\vec{v}_{\mathrm{T}} = \vec{v}_{\mathrm{thumb}} = \vec{p}_{\mathrm{tip}}^{\mathrm{thumb}} - \vec{p}_{\mathrm{knuckle}}^{\mathrm{thumb}}$$

$$\theta = \arccos\left(\frac{\vec{v}_{\mathrm{I}} \cdot \vec{v}_{\mathrm{T}}}{\|\vec{v}_{\mathrm{I}}\| \, \|\vec{v}_{\mathrm{T}}\|}\right)$$

Given just the L part of this gesture, two green spheres will appear on the tip of the user's index and thumb respectively. This will indicate that a frame is cued and ready to spawn into the scene. With this cue, a tap of the thumb and index finger of the opposing hand will generate the view frustum.

4 IMPLEMENTATION DETAILS

4.1 Shot Planning

Shot frames are 3D models ("entities") loaded from RealityKitContent into the scene. To ensure consistent placement and shot organization, an origin entity was created to act as a custom world origin users could manipulate and use to re-align loaded plans to what they see. We attempted to use the built-in ARKit world tracking system for consistent placement, however we found it's inconsistency undesirable and reverted back to a manual placement system. Due to the limitations of how RealityKit manages interactions, the shot frame entities are added directly to the world root and when edits occur to the custom origin entity all frames are temporarily parented to it to be properly transformed along with it.

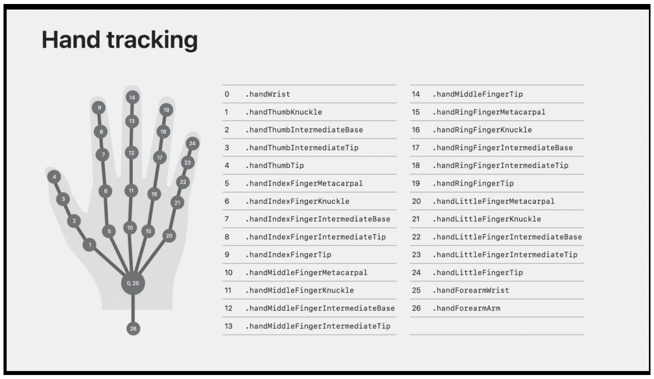
The Apple Vision Pro's interaction model consists of using the user's eyes to focus on subjects then using hands movements to interact. To interact with frames, users can look at any part of a frame and pinch their fingers to "grab" onto it. With a one handed pinch and drag, users can move a frame around in 3D space. Users can pinch and grab with both hands to enable scaling and rotation by orbiting their hands. The frame will move with accordance with the user's hand movement. Because the Apple Vision Pro uses eye tracking, users do not have to physically approach a frame to move it and can instead look at it from any distance and manipulate it. This enables efficient planning and quick adjustments. The same interaction model is used to control the custom origin world origin point. The origin is intended to be a fixed object and used as a reference coordinate for the scene. Therefore is pinned to "ground" level (it does not translate vertically into the air) and rotates around the vertical axis.

A control panel accompanies each frame. This control panel holds the shot's name (the first digit is the scene name and the following letter is the shot name). Users can use the stepper interface to increment the shot's name. It will automatically pick the next available shot name to avoid duplicate names. Following the shot name is the orientation information broken out as input fields for fine-tuned controls. If the user is satisfied with the frame's position, they can also lock it in place to prevent accidental changes. At the bottom, users can add additional notes, much like they can on a storyboard frame.

When a new frame is added, the user's head pose is captured and used as the initial spawn pose of a frame entity. We then shift the frame slightly forward and down to match the eye-line of the user, enabling them to place shots where they were looking. When combined with the handtracked gestures, this creates an intuitive and seemless interface for rapidly placing shots for planning scenes.

4.2 Handtracking

Upon initialization of our scene, hand tracking components would be mapped to two new entities representing each respective hand chirality. For each entity, a series of joint trackers would be established for all the joints contained within the hand. In early development we mapped purple spheres to each bone joint's location for visual representation of the hand tracking state. In current iteration, two faintly visible spheres are mapped to the joints associated with L gesture and frame spawning; the rest do not have models, but could easily receive them for additional feature/gesture implementations. The next step was driving frame spawning via custom gesture recognition. Here, the custom gesture we use for spawning in new frames is forming a tight vertical L shape with one hand's thumb and index finger, then tapping the thumb and index finger of the opposite hand. A frame will then spawn ~0.6 meters in front of the user's direct gaze vector, with the corresponding orientation of the headset. The engagement of the L gesture is determined by calculating: 1)Distance vectors from the thumb knuckle (1) to thumb tip (4) and index finger metacarpal (5) to index finger tip (9). (2)Dot product of these vectors. (3)If the degrees between the vectors(inverse cosine of dot product) is within an adjustable threshold(set by default to 55°), a flag indicating the engagement of the L gesture is toggled on until disengagement. When the L gesture is engaged, the (1) and (4) joint models turn solid green to indicate the gesture is cued and a frame is ready to be spawned upon tap gesture of the other hand.


Hand tracking

0	.handWrist
1	.handThumbKnuckle
2	.handThumbIntermediateBase
3	.handThumbIntermediateTip
4	.handThumbTip
5	.handIndexFingerMetacarpal
6	.handIndexFingerKnuckle
7	.handIndexFingerIntermediateBase
8	.handIndexFingerIntermediateTip
9	.handIndexFingerTip
10	.handMiddleFingerMetacarpal
11	.handMiddleFingerKnuckle
12	.handMiddleFingerIntermediateBase
13	.handMiddleFingerIntermediateTip
14	.handMiddleFingerTip
15	.handRingFingerMetacarpal
16	.handRingFingerKnuckle
17	.handRingFingerIntermediateBase
18	.handRingFingerIntermediateTip
19	.handRingFingerTip
20	.handLittleFingerMetacarpal
21	.handLittleFingerKnuckle
22	.handLittleFingerIntermediateBase
23	.handLittleFingerIntermediateTip
24	.handLittleFingerTip
25	.handForearmWrist
26	.handForearmArm

Fig. 2. Apple's defined hand joints mapped to specific indices. [Varrall 2023]

4.3 Scene data

Internally, each shot is defined by a serializable model that tracks it's name, transformation, additional notes, and other interface details such as if it has been initialized yet. The shots are organized in an array in the main app's state, under the active scene. During saving and loading, the array of shot models is serialized into a JSON file and saved to the application's documents directory. From there it persists across application launches and can be reloaded in any location. As a simple JSON file, it can also be shared using the default Apple share sheet to other devices that can parse the data and produce other files. All of the files and data are managed locally and do not require an internet connection or online storage to operate.

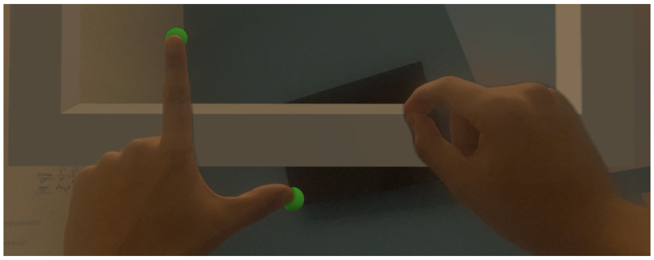


Fig. 3. L gesture with opposite hand press spawning in a frame entity.

5 DISCUSSION OF BENEFITS AND LIMITATIONS

This section may end up being combined with Sections **??** and 6. The goal is to outline key benefits and limitations. Ideally, you'd link this back to the fundamental details of your approach in Section 3. But, some of these limitations may result from your implementation itself. Try to make conclusions about your approach and why it might have advantages and disadvantages.

Our largest contribution was enabling users to plan storyboard-like shots using their exact viewing perspective on their shooting location. Specifically, the ability for a user to look in the direction of a shot and use a hand gesture to place a 3D frame, and then observe and manage all shots in a given scene is a novel feature we introduce to the preproduction pipeline using mixed reality. Through our discussion with field expert Nathan Matsuda we believe this is a beneficial feature to the industry.

The Apple Vision Pro provides a solid platform for mixed reality experiences which are very helpful for shot planning. Users can see through to their set and also to their surroundings. Using mixed reality also means users will be planning on their actual set location instead of a virtual approximation, often used by existing VR-based solutions.

The plans generated by our software encode the shot locations and orientation and can be exported in a generic format that other software can use. In preproduction, many people not on set (such as producers and executives) may need to see and understand what is happening in the plan. Through the JSON export, planning information can be extracted and converted into alternate forms that can be passed around.

We are limited by our constraints of using the Apple Vision Pro. Some limitations of the hardware platform include it's reliance on camera based tracking and ergonomic limitations. Because the Apple Vision Pro is unable to track its position in the dark, our application is unable to maintain consistent positioning. If the device is unable to see the users hands, the user likewise is unable interact with the shot frames. The Apple Vision Pro is also not particularly comfortable to use for long durations which can hinder the creative process of a user.

6 FUTURE WORK

While we are excited for the features we present, there are even more features we can bring to further flush out StoryboardXR.

One feature, as recommended by Matsuda, is to generate traditional storyboard-like PDF exports of the 3D plans that can be easily shared. We currently have the intermediate data, JSON files with shot orientation information. To achieve 2D storyboard exports, we would need to get a virtual scan of the set to merge with our scene plan. We could then use virtual cameras placed at each of the shot frames to recreate the planned view from a virtual plan.

Other features we are excited about include the ability to plan for camera movements and transitions between shots. Currently the shots are numbered but there aren't visual guides that help spatially represent transitions between shots. This could be an additional and novel area of improvement to the traditional storyboard system that is made possible through mixed reality.

Another exciting feature of mixed reality is the potential to share experiences with multiple people. Planning on a set involves many people and a useful feature this project could include would be the ability to allow multiple users with multiple headsets to interact with the same scene plan simultaneously. This could help directors, preproduction artists, and cinematographers to collaborate seamlessly on the same set.

7 CONCLUSION

We believe mixed reality can play an important role in film preproduction and will be a platform to accelerate the creative work of professionals alongside other virtual reality and previsualization tools used by filmmakers today. Our intuitive interface for shot planning and sharing lowers the barrier of entry for new creatives to plan out complex multi-shot scenes, enable professionals to work more efficiently, and enable new methods for preproduction teams to share their plans between crews. We are also looking to introduce practical tools to the Apple Vision Pro as this field of mixed reality headsets are becoming more available to consumers. We look forward to improving our platform's capabilities alongside improvements to hardware in this field.

ACKNOWLEDGMENTS

Big thanks to Doug Lanman, Evan Zhao, Shaan Chattrath, John Akers, and anyone else that made CSE 493V possible. Additional thanks to Nathan Matsuda for giving relevant industry input on real-world use cases.

REFERENCES

Hye-Young Jo, Ryo Suzuki, and Yoonji Kim. 2024. CollageVis: Rapid Previsualization Tool for Indie Filmmaking using Video Collages. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 164, 16 pages. https://doi.org/10.1145/3613904.3642575

Moviestorm. 2024. FirstStage. https://firststage.moviestorm.co.uk/.

Spenser Sakurai. 2022. How to STORYBOARD in VR. HowtoSTORYBOARDinVR

StudioBinder. 2025. StudioBinder. https://www.studiobinder.com/.

Tvori. 2025. Tvori. https://tvori.co/.

Stuart Varrall. 2023. Hand tracking in VisionOS. https://varrall.substack.com/p/hand-tracking-in-visionos

All code is open-source on GitHub: https://github.com/StoryboardXR/StoryboardXR