Latest Manual URL

https://docs.google.com/document/d/1slQ6CiH0NkkO8xIbc81xAmmyzbnSbz9fnlUemrBLRUo/

Technical Support URL

Discord: https://discord.gg/6UGu7sy

Revision History

v1.2.0

- Bug Fixes:
  - Avoids null references when recreating cameras
  - Removed more GetComponents and replaced with direct references for more efficiency
- Added Horizontal and Vertical Perspective Support
- Added warning messages when Game window does not match projector resolution
- New Qoobit Discord Support Channels added. Please join at: https://discord.gg/6UGu7sy

v1.1.0

- Bug Fixes:
  - Some fields not being serialized because of editor changes not dirtying scene
  - Loaded calibrations not properly being initialized into UI
  - Control point vertex calculations for edge cases
- Rebuilt code without requiring Substance textures. We had been having a lot of feedback with different versions of Substance not working with Unity 2018, so we reimplemented the shader as a standalone not requiring Substance. This means that in this update, it will be backwards compatible with Unity 2017. Adjusting fades should be more responsive and less buggy with this new implementation.

v1.0.5

- Added hard limits to the **Projection Warp System** editor to prevent inputting unsupported values
- **Overlap** and **Projection Camera Space** are also hidden if there is only 1 camera being used
- Major code refactoring for increased efficiency removing many instances of *GetComponent*
- Moved many prefabs and substances to the resources folder so that they can be referenced and instantiated in the new workflow behind the scenes
- Added version number to the calibration json in case future versions have different formats

- Added sample scene for use with Unity Video Player.

v1.0.4

- Major update to support Substance in Unity for Unity 2018+. Primarily removed the use of deprecated **ProceduralMaterial**.

v1.0.3

- Updated **Control Point** and **Projection UI** prefab to be initially disabled to prevent initial blip when launching from build.

v1.0.2

- Added support to launch a default json configuration file if it is located in the same directory as the exe. The filename must match **default_calibration.json**.

v1.0.1

- Minor bug fixes

v1.0.0

- Initial submit

What does it do?

This Unity Asset allows you to manage multiple unity cameras to produce large, seamless horizontal or vertical projection panoramas with one controller. It can be used for flat projection surfaces or curved panorama surfaces.

Prerequisite

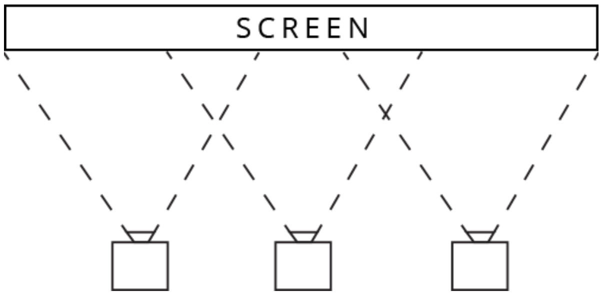
The plugin requires at least one projector, with all projectors having the same resolution settings. It is recommended that all projectors used are of the same model for best color matching.

Supported Projector Configurations

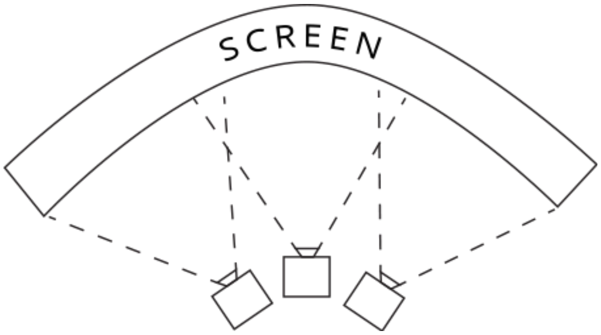
Place the projectors at a distance where the **image slightly overshoots the projected surface**. If possible, **try to align the top or bottom edge** to the projection surface for minimal tweaking for best results. Also, make sure that there is at least 10% overlap between each projected image. Be sure to reorder the cameras in sequence from left to right inside the display settings.

Horizontal Panorama

Bird’s Eye View



Horizontal Line

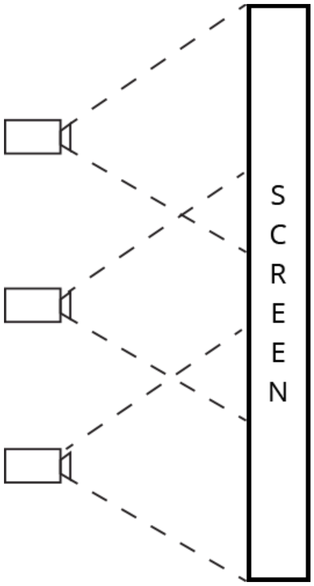


Horizontal Curved

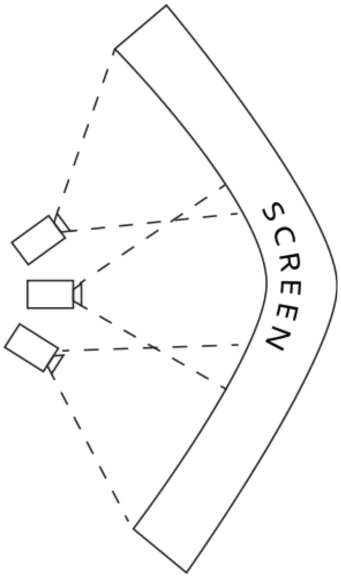
The projectors need to be lined up either horizontally in a straight line or curved with adequate overlap between each projector image on the left and right.

Vertical Panorama

Side View



Vertical Line



Vertical Curved

The projectors need to be lined up either vertically in a straight line or curved with adequate overlap between each projector image on the top and bottom.

How it Works

We’ve scripted a system prefab for drag and drop use in your scene. The **ProjectionWarpSystem** script attached to the prefab controls the shared properties of all the cameras being used.

Please view our tutorial video for the functions of this version:
https://www.youtube.com/watch?v=_6Qgq60m8DI

We begin by entering the basic global settings that will be using for the setup.

ProjectionWarpSystem Component Values

Arrangement	Are the projectors horizontally or vertically placed in the physical world? Is what you’re trying to draw using an orthographic or perspective camera?
UI Toggle Key	Key to use to show/hide the calibration UI at runtime
Projector Resolution	The resolution settings on the projectors. Be sure that these are synced across all projectors (and Game windows).
Camera Count	The number of projectors being used.
X Divisions/Y Divisions	The number of divisions to use for the render target for shape adjustments.
Overlap	The amount of overlap between each camera. This will reduce the overall area of the panorama, but is necessary for making seamless blend transitions. In HORIZONTAL_ORTHOGRAPHIC and VERTICAL_ORTHOGRAPHIC arrangements, this refers to the amount of positional offset for each of the cameras to create the overlap. In HORIZONTAL_PERSPECTIVE And VERTICAL_PERSPECTIVE, this refers to the amount of angular offset for each of the cameras to cover the desired field of view
Field of View (Perspective Only)	Field of view you wish to cover with the system. In HORIZONTAL_PERSPECTIVE, the value represents the

	horizontal field of view angle you wish to cover in your scene. In VERTICAL_PERSPECTIVE, the value represents the **vertical field of view** angle you wish to cover.
Near	Near clipping plane for all pano cameras
Far	Far clipping plane for all pano cameras
Projection Camera Spacing	The amount of space between each render texture mesh. If 0 is used, it is advised to set layer masks for Projector Image.

NOTE When calibrating, please remember to check the **Show Projector Warp GUI** toggle to display the UI for making adjustments. It will only be shown on display 1 because Unity UI has bugs dealing with mouse positions on multiple displays. You can also map a keyboard key to toggle this on and off.

TIP Individual camera tweaks can be made by selecting each GameObject under **Projection Images' Projection Mesh** component. They are stored on a per camera basis and used for the purposes of lining up the projector image for seamless blends between projectors.

Projection Mesh Component Values

Pano Camera Offset	Arbitrary X and Y offsets for the pano camera added on top of the Overlap values (Reset to 0 if using perspective cameras)
Left Fade Range	Shader Mask Left Fade Range
Left Fade Choke	Shader Mask Left Fade Choke
Right Fade Range	Shader Mask Right Fade Range
Right Fade Choke	Shader Mask Right Fade Choke
Top Fade Range	Shader Mask Top Fade Range
Top Fade Choke	Shader Mask Top Fade Choke
Bottom Fade Range	Shader Mask Bottom Fade Range

Bottom Fade Choke	Shader Mask Bottom Fade Choke
Trapezoid Anchor	Anchor of trapezoid shape warp
Trapezoid Angle	Angle relative to the anchor above
Skew Anchor	Anchor of skew shape warp
Skew Angle	Angle relative to the anchor above
Top Offset	The X and Y top control point offset
Bottom Offset	The X and Y top control point offset

NOTE Trapezoid warps are not necessary if the projector has the Keystone feature. Keystone will align the left and right edges of the image automatically, meaning it's not necessary to use this feature. If using VERTICAL_PERSPECTIVE, be sure to turn off keystone and manually adjust the trapezoid value for better results.

After setting up the parameters above, a render texture will be generated for each pano camera.

Saving and Loading Calibrations

A JSON file is used to save and load the calibration settings for all the cameras in the system.

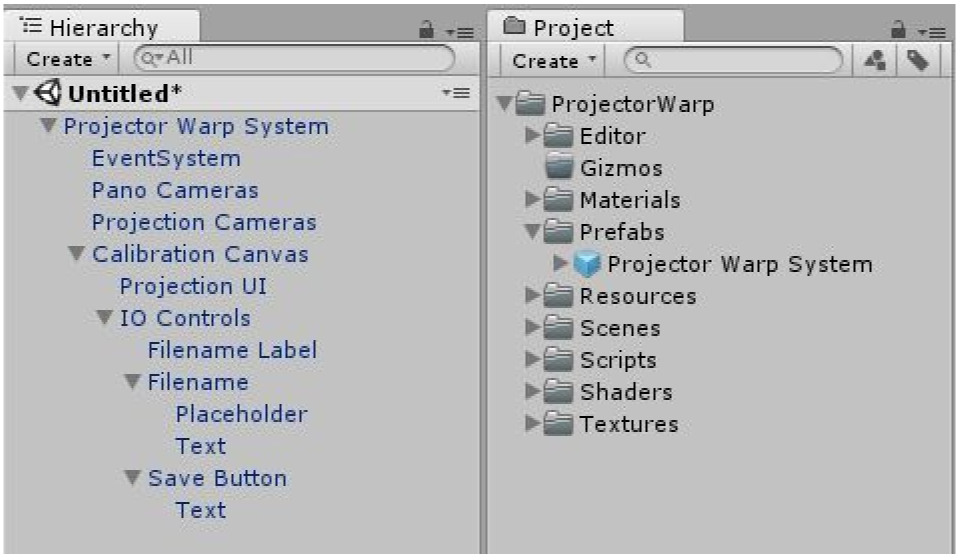
To save out the calibration file, goto the **ProjectionWarpSystem** component and click the **Save Calibration** button. You will be prompted with a dialog to save the file.

To load a calibration file, goto the **ProjectionWarpSystem** component and click the **Load Calibration** Button. You will be prompted with a dialog to search for a previously saved JSON configuration file.

Pipeline

1. Set the physical projectors in the desire arrangement and try to lock them in place so they don't move around. Synchronize the settings across the projectors and make sure they have adequate overlap in the physical space.
2. Drop the **Projection Warp System** prefab into the current scene and position the GameObject into the region you wish to use for the projection. Animations should be

done only on the **Projection Warp System** prefab. All other transforms underneath are being controlled by scripts.



3. Enter the global settings for the cameras. Begin by selecting the camera arrangement.

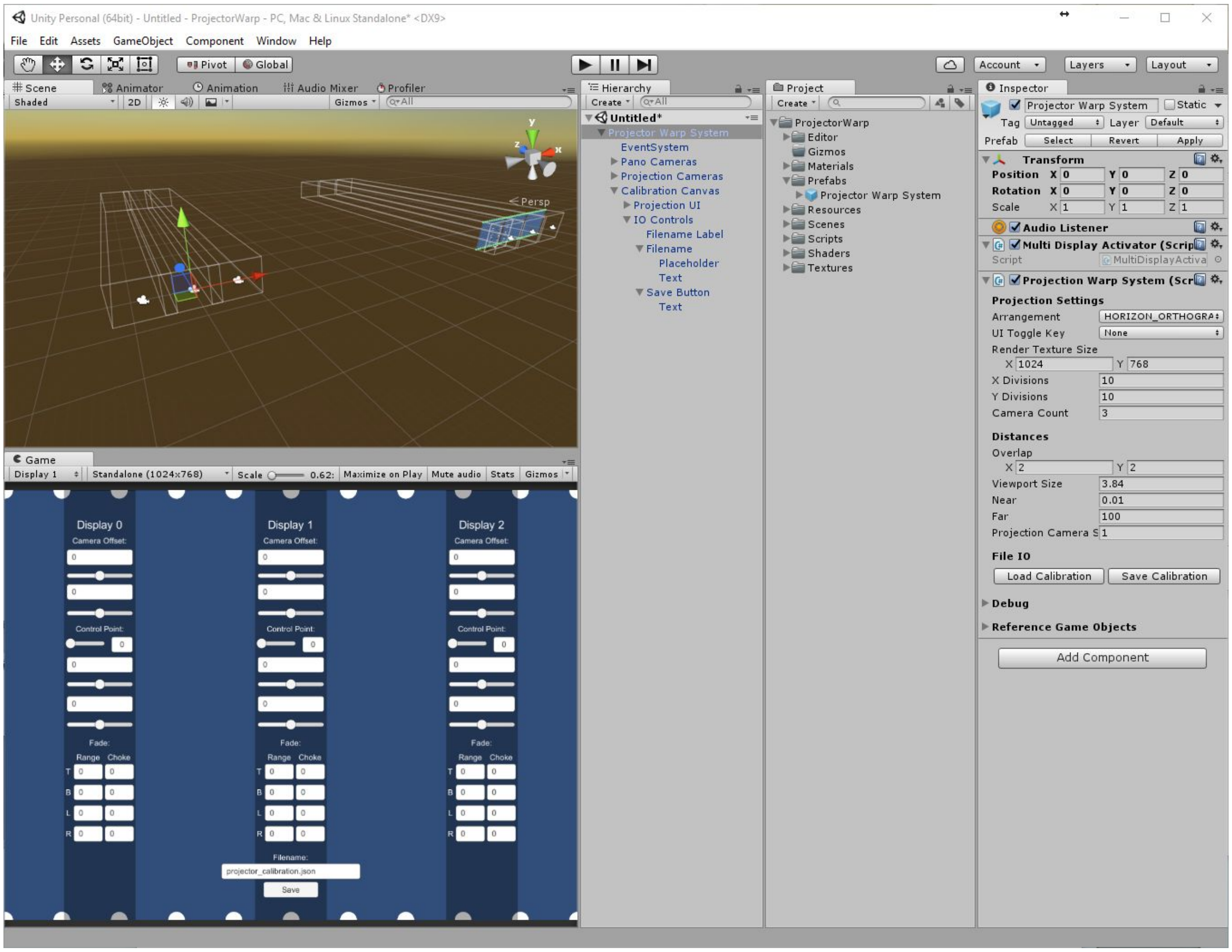
   The **Projector Resolution** should be set to the exact resolution of the projectors being used. By default they have been set to 1024x768. Be sure to set all your Game windows ot the same resolution or you will see a warning.

   The **X Divisions** and **Y Divisions** are arbitrary determined by you and should be set to a resolution that suitably represents the projection surface. The more divisions you have, the more manual tweaking will be needed on the control points at the later stages. However, with more control points, you will have a more accurate deformation.

   Camera Count needs to be set to the number of projectors being used. They will be centered to the parent prefab.

   Near and **Far** are the clipping planes that will be used for all the Orthographic Cameras.

   Overlap is for setting the amount of overlap region between each camera in the Panorama. Remember that in ORTHOGRAPHIC modes these are positional offsets and in PERSPECTIVE modes these are rotational offsets. Make sure that a non zero positive value is set for best results. A good point of reference is to try and get at least 10% of the image overlapped.



Projector Warp System ☐ Static

Tag Untagged Layer Default

Prefab Select Revert Apply

Transform

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

☑ Audio Listener

☑ Multi Display Activator (Script)

Script MultiDisplayActiva

☑ Projection Warp System (Script)

Projection Settings

Arrangement	HORIZON_ORTHOGRA
UI Toggle Key	None
Render Texture Size	X 1024 Y 768
X Divisions	10
Y Divisions	10
Camera Count	3

Distances

Overlap	X 2 Y 2
Viewport Size	3.84
Near	0.01
Far	100
Projection Camera S	1

File IO

Load Calibration Save Calibration

Debug

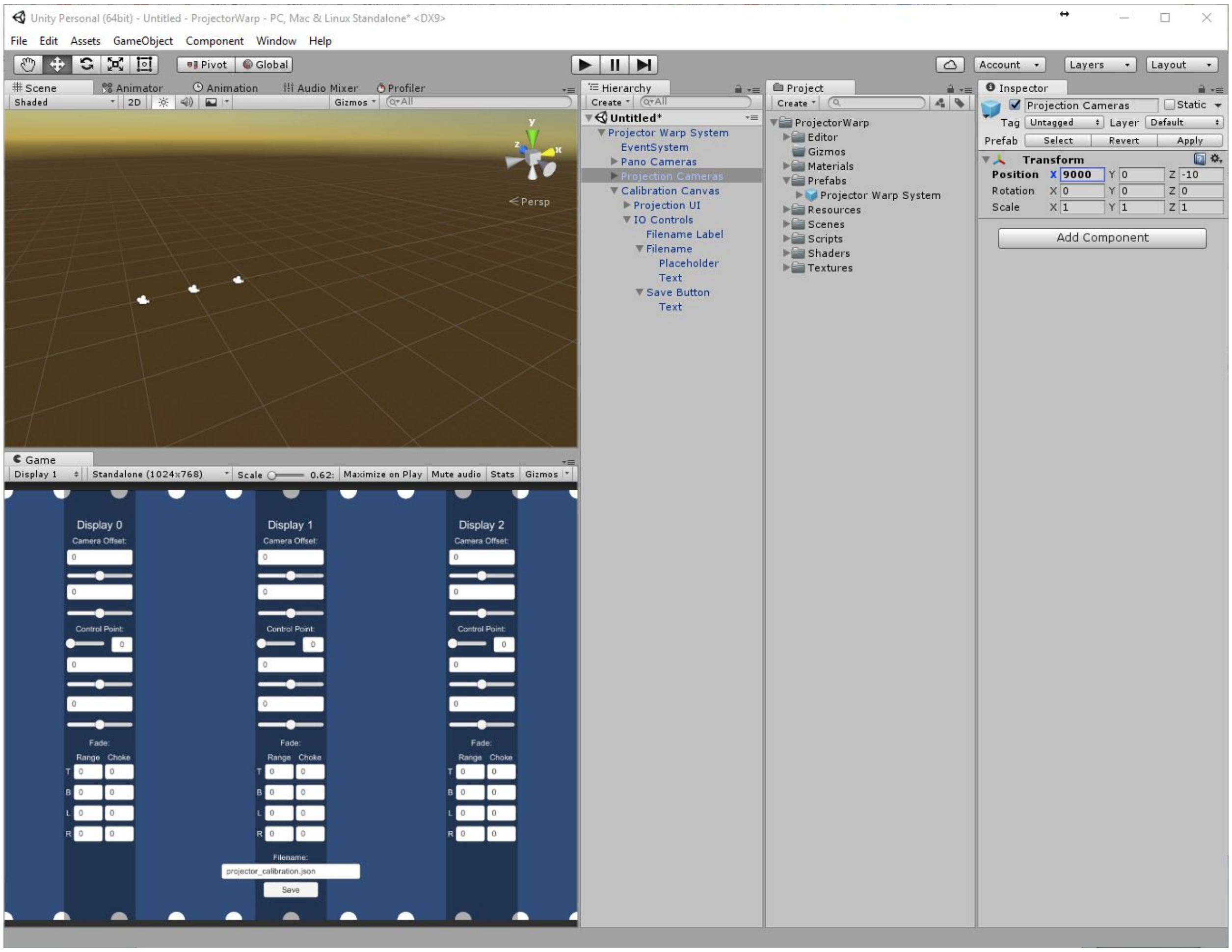
Reference Game Objects

Add Component

WARNING	After these global values have been set, avoid making changes to them since they will re-generate the meshes for the projections. This includes, but is not limited to, changing the X Divisions, Y Divisions and Camera Count.

4. After setting the values above, each camera will be rendered to their own **Render Texture** which are then projected onto a deformable plane listed under **Projection Cameras**. Since these planes are rendered onto different displays, it is a good idea to move them out of the viewing range of the **Pano Cameras**. To move them, just move the **Projection Cameras** GameObject.

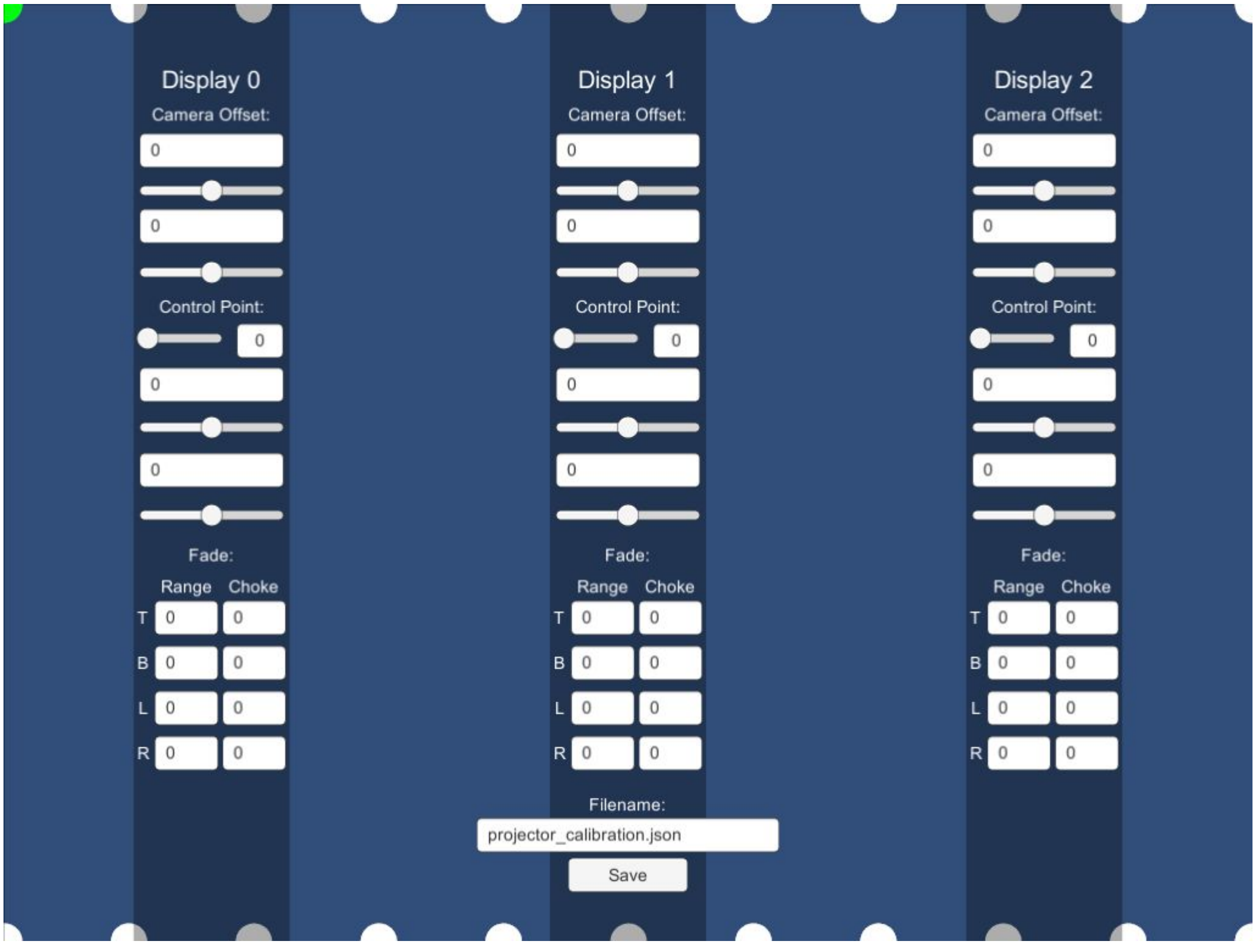
   In the event, that they cannot be moved out of the viewing range of the **Pano Cameras**, create layers and assign them to the **Projection Image** GameObjects so that they will not be rendered through the **Pano Cameras**.



Projection Cameras ☐ Static
Tag Untagged Layer Default
Transform
Position X 9000 Y 0 Z -10
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

The **Projection Camera Space** can also be used to space apart the generated mesh planes so that their images do not overlap on the cropped out areas of the displays.

5. At this point we need to build out and run the program since the Unity Editor doesn’t allow simultaneous displaying of multiple screens during play mode. In editor mode, you can still swap between displays, but it’s not very useful for trying to line up the images. After building and running, your screens will be displayed with multiple white balls representing the control points on each projector.



Display 0

Camera Offset: 0, 0

Control Point: 0

0, 0

Fade:

	Range	Choke
T	0	0
B	0	0
L	0	0
R	0	0

Display 1

Camera Offset: 0, 0

Control Point: 0

0, 0

Fade:

	Range	Choke
T	0	0
B	0	0
L	0	0
R	0	0

Display 2

Camera Offset: 0, 0

Control Point: 0

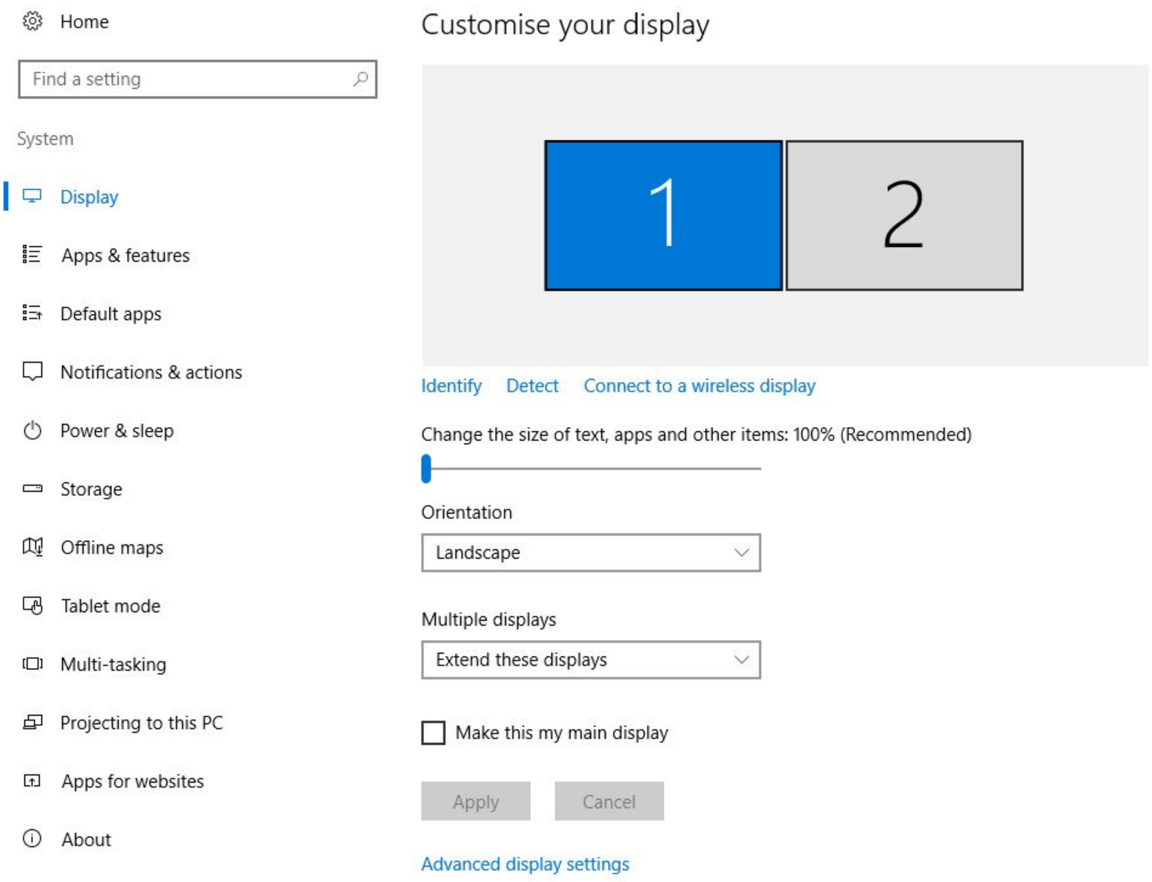
0, 0

Fade:

	Range	Choke
T	0	0
B	0	0
L	0	0
R	0	0

Filename: projector_calibration.json

Double check that all displays are in the correct sequence. If not goto the display settings and rearrange them accordingly.



Customise your display

Change the size of text, apps and other items: 100% (Recommended)

Orientation: Landscape

Multiple displays: Extend these displays

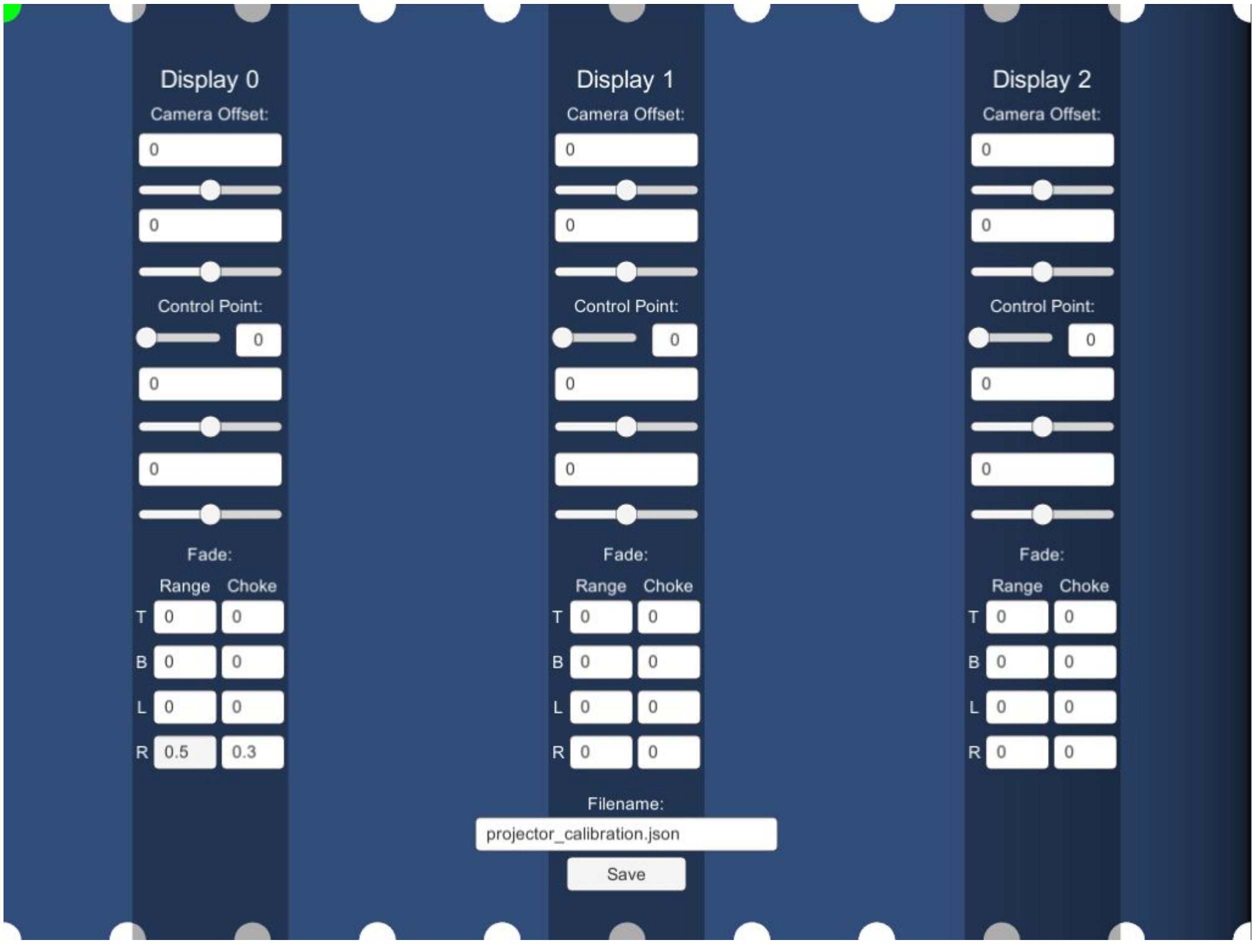
□ Make this my main display

The controlling UI has all been consolidated on display 1 since Unity UI doesn't work properly beyond display 1. Scroll through the control points and adjust their positions to match the projection surface. The control points can also be clicked for faster selection.

TIP Try to physically move the projectors to match a general point of reference for the horizon. This usually means trying to match the images so that the top or bottom edge lines up with the projection surface as much as possible. Slightly overshooting the surface with the projector images is recommended since you can pull the image back inside the viewing range.

6. After lining up the images properly, it should be looking pretty good with minimal to no blurs in the overlapping regions. Except now, the overlapping regions will be displaying overly bright. We will address that in this step. By using the **Fade** parameters at the bottom of the UI to adjust the fade regions. There will be a **range** and **choke** value that

can be adjusted to create a desired gradient. These values are linked to a substance that is responsible for procedurally generating the alpha gradient used for the blend.



Display 0

Camera Offset: 0, 0

Control Point: 0, 0, 0

Fade:

	Range	Choke
T	0	0
B	0	0
L	0	0
R	0.5	0.3

Display 1

Camera Offset: 0, 0

Control Point: 0, 0, 0

Fade:

	Range	Choke
T	0	0
B	0	0
L	0	0
R	0	0

Display 2

Camera Offset: 0, 0

Control Point: 0, 0, 0

Fade:

	Range	Choke
T	0	0
B	0	0
L	0	0
R	0	0

Filename: projector_calibration.json

7. SAVE!

The most important step! After making the adjustments above, it would be a real pain if you forgot to save (since changes made at runtime are not saved automatically). Clicking the **Save** button will store all the changes into a JSON file that can be used later. Double check that the JSON has been saved and exists in the proper location and close the executable.

8. Go back to Unity and click the **Load Calibration** button and select the JSON file that was just exported. This will bring the settings in as a default and you're good to go. Any future builds will utilize these calibration settings.

TIP If you’re running the exe on a computer without Unity and need to update the calibrations, rename your calibration filename to **default_calibration.json** and keep this file in the same directory as your export executable. If the plugin finds this file at startup, it will opt to use this instead of the provided json file.

Potential Future Upgrades

- Cube Projector Mapping
- SteamVR Integration

If you have any other features you would like to suggest or if you find any bugs or problems, please send us an e-mail at info@qoobit.com or visit us at our Discord channel for more immediate responses: https://discord.gg/6UGu7sy

Thanks for using our asset!