Easy Collider Editor

Table of Contents

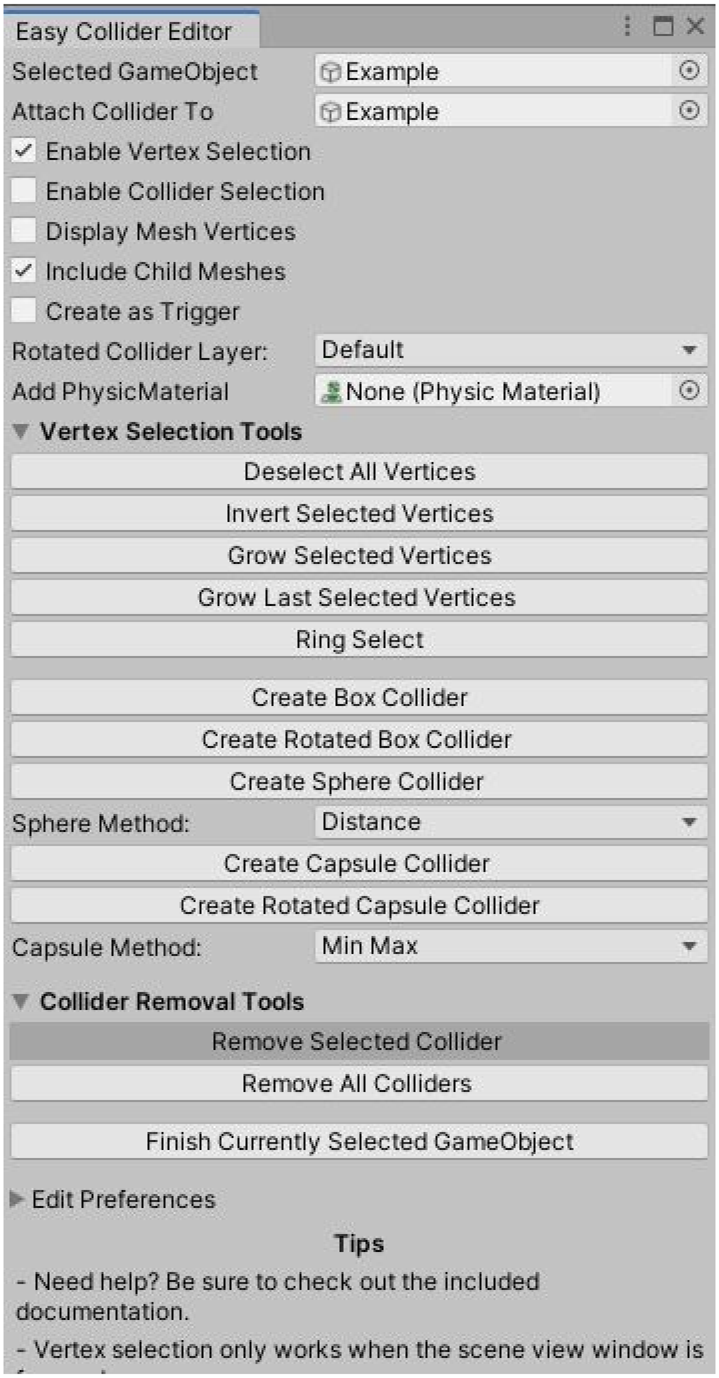
Quick Start Guide

Just want to jump in and make some colliders? Here's a simple guide to the basics.

1. Open the editor window from Window => Easy Collider Editor
2. Drag a GameObject from the scene into the Selected GameObject field.
3. Enable the Vertex Selection toggle.

4. Click into the SceneView and move the mouse over the Selected GameObject.
5. Press V to select vertices on the mesh, and B to select any point on the mesh.
6. Click the Create Collider Button.
7. When done creating colliders, click the Finish Currently Selected GameObject button.

Editor Window UI



Selected GameObject: Example

Attach Collider To: Example

☑ Enable Vertex Selection

□ Enable Collider Selection

□ Display Mesh Vertices

☑ Include Child Meshes

□ Create as Trigger

Rotated Collider Layer: Default

Add PhysicMaterial: None (Physic Material)

Sphere Method: Distance

Capsule Method: Min Max

Tips

- Need help? Be sure to check out the included documentation.

- Vertex selection only works when the scene view window is

General UI

In general, hovering over the text of the UI when the window is in focus will provide tooltips that describe the options as well.

When you first start the asset, you will notice that some of the buttons appear to be disabled. Hovering over the disabled buttons will give you a tooltip on why the button is currently disabled. Buttons that require certain things to work are disabled and displayed differently when those conditions are not met, and enabled when they are met.

From top to bottom in the image above:

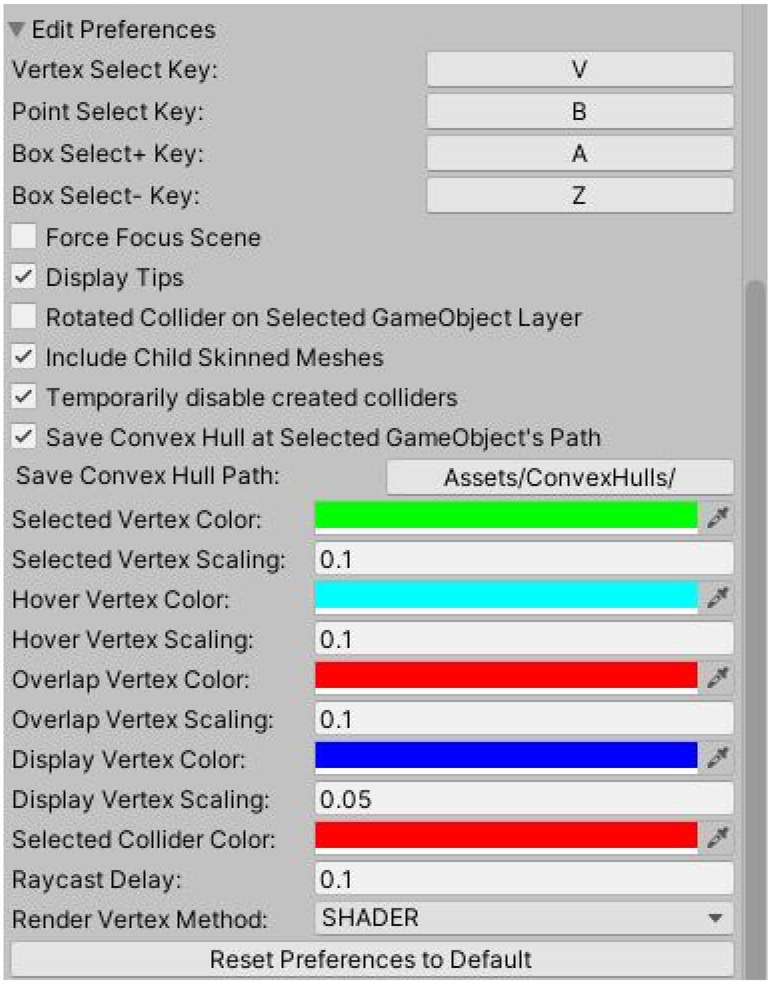
- **Selected GameObject:** the gameobject you wish to be able to select vertices from, or the parent of all the gameobjects you wish to select vertices from.
- **Attach Collider To:** the gameobject you wish the created collider to be attached to. This gameobject is also used when calculating the collider, as it uses this gameobjects local space. This is especially useful for things like creating colliders on a skinned mesh, as you can select the whole mesh, but attach the colliders to each individual bone.
- **Enable Vertex Selection**: Enables selection of points and vertices in the scene view.
- **Enable Collider Selection:** Enables collider selection in the scene view
- **Display Mesh Vertices:** Displays all selectable vertices. Generally used to make sure meshes are being detected properly, or that the shader or gizmos are functioning correctly.
- I**nclude Child Meshes:** When enabled, points and vertices on meshes from the Selected GameObject's children are also able to be selected.
- **Create as Trigger:** When enabled, creates the collider with the IsTrigger property checked.
- **Rotated Collider Layer:** The layer to set on the gameobjects created for rotated colliders. This option only displays if the Rotated Collider on Selected GameObject Layer toggle in preferences is disabled.
- **Add PhysicMaterial:** Sets this physic material to any created collider's PhysicMaterial field
- **Vertex Selection Tools:** Foldout to display vertex selection tools.
    - **Deselect All Vertices:** Deselects all currently selected vertices.

  - **Invert Selected Vertices:** All selected vertices get unselected, and all unselected get selected.
  - **Grow Selected Vertices:** Expands the selected vertices along triangle edges of all selected vertices.
  - **Grow Last Selected Vertices:** Expands the selected vertices along triangle edges of only the last selected vertices.
  - **Ring Select:** Attempts to select vertices around the mesh by looping around the mesh following triangle edges.
- **Create Box Collider:** Creates a box collider from the selected points.
- **Create Rotated Box Collider:** Creates a gameobject that is aligned with the first 3 points selected, attaches it to the Attach Collider To gameobject, and then creates a box collider on the created object.
- **Create Sphere Collider:** Creates a sphere collider using the algorithm specified by the Sphere Method field
- **Sphere Method:** The algorithm to use when generating a sphere collider
- **Create Capsule Collider:** Creates a capsule collider using the algorithm specified by the Capsule Method field.
- **Create Rotated Capsule Collider:** Creates a gameobject that is aligned with the first 3 points selected, attaches it to the Attach Collider To gameobject, and then creates a capsule collider on the created object.
- **Capsule Method:** The algorithm to use when generating a capsule collider.
- **Create Convex Mesh Collider:** Creates mesh from all the points that are selected, saves it in the location specified in preferences. This mesh is then used to generate a convex mesh collider, and is attached to the Attach Collider To gameobject.
- **Collider Removal Tools:** Foldout to display capsule removal buttons
- **Remove all Colliders:** Removes all colliders on the Selected GameObject, and children if Include Child Meshes is enabled.
- **Remove Selected Collider:** Removes the currently selected collider
- **Finish Currently Selected GameObject:** Finishes editing on the Selected GameObject. This ensures proper cleaning up of added components required for this asset. **It is**

particularly important to use this before closing unity, or before entering or exiting prefab isolation mode.

- **Edit Preferences:** This is a foldout that can be used to display all the various preferences options.

Preferences UI



▼ Edit Preferences

Vertex Select Key: V

Point Select Key: B

Box Select+ Key: A

Box Select- Key: Z

☐ Force Focus Scene

☑ Display Tips

☐ Rotated Collider on Selected GameObject Layer

☑ Include Child Skinned Meshes

☑ Temporarily disable created colliders

☑ Save Convex Hull at Selected GameObject's Path

Save Convex Hull Path: Assets/ConvexHulls/

Selected Vertex Color: 🟩

Selected Vertex Scaling: 0.1

Hover Vertex Color: 🟦

Hover Vertex Scaling: 0.1

Overlap Vertex Color: 🟥

Overlap Vertex Scaling: 0.1

Display Vertex Color: 🟦

Display Vertex Scaling: 0.05

Selected Collider Color: 🟥

Raycast Delay: 0.1

Render Vertex Method: SHADER

In order from top to bottom of the preferences foldout UI:

- **Keys:** To change a key, press the button and then press a key on the keyboard.
  - **Vertex Select Key:** Key to press to select a vertex.
  - **Point Select Key:** Key to press to select a non-vertex point on the mesh.
  - **Box Select+ Key:** Key to hold down prior to a click and drag motion to only select the vertices in the box. Note that setting this or the Box Select- Key to an

EventModifiers key (https://docs.unity3d.com/ScriptReference/EventModifiers.html) can cause issues, and will only work if held down after starting a box selection.

  - **Box Select- Key:** Key to hold down prior to a click and drag motion to only deselect the vertices in the box.

- **Force Focus Scene:** Force focuses the scene view when either vertex, or collider selection is enabled. Allows selection without having to click into the scene view first. Can make it difficult to move, or change values in other inspectors.

- **Display Tips:** When enabled, displays helpful tips for common issues at the bottom of the window.

- **Rotated Collider on Selected GameObject Layer:** When enabled, rotated colliders automatically get created on the same layer as the Selected GameObject's layer.

- **Include Child Skinned Meshes:** Includes skinned meshes on children when Include Child Meshes is enabled.

- **Temporarily disable created colliders:** Sets all created colliders as disabled until Finish Currently Selected GameObject is pressed, then enables them. Prevents created colliders from interfering in point and vertex selection.

- **Save Convex Hull at Selected GameObject's Path:** When creating a convex mesh collider, attempt to save the mesh at the same location in the project as the selected gameobject's path.

- **Save Convex Hull Path:** This folder is used to save the mesh collider's mesh if Save Convex Hull at Selected GameObject's Path is disabled, or trying to find the path of the Selected GameObject fails.

- **Vertex Color:** The color used when drawing each type of vertex below in the scene view.

- **Vertex Scaling:** The size of each displayed vertex type, see below for descriptions.

  - **Selected Vertices:** These are the vertices or points you have selected.

  - **Hovered Vertices:** These are the vertices or points that are being hovered over that will be selected if the key is pressed.

  - **Overlapped Vertices:** These are the vertices or points that are currently hovered over that will be deselected if selected again.

    - **Displayed Vertices:** These are the vertices drawn when the Display Mesh Vertices toggle is enabled.

- **Selected Collider Color:** The color to use when drawing the selected collider.

- **Raycast Delay:** The delay to use when raycasting, and updating the box select vertices.

- **Render Vertex Method:** The method to use to render selected, hovered, and overlapped vertices. The SHADER method is significantly faster, but requires your system to be able to use compute buffers. See https://docs.unity3d.com/Manual/class-ComputeShader.html for more details.

- If the Render Vertex Method is GIZMOS the following options are available:

    - **Draw Gizmos:** Allows you to toggle on and off the drawing of gizmos. As gizmos are extremely slow compared to a shader, they can cause significant slowdown when a large amount of points are selected.

    - **Gizmo Type:** The type of gizmo to draw, either Spheres or Cubes.

    - **Use Fixed Gizmo Scale:** When enabled, gizmos use a fixed screen size.

- **Reset Preferences to Default:** Resets all the preferences above to their default values.

Selecting Vertices and Points

This new version of Easy Collider Editor contains many more ways to select points on your mesh. It also makes it easier to select vertices by improving undo/redo functionality. All vertex selection methods can be undone/redone by the usual undo/redo shortcuts and buttons in unity.

All selections update and display in colors and sizes set in the preferences options. If you find the boxes are too big for your mesh, try scaling them down.

By default:

- Vertices and points that will be selected are colored in light blue
- Vertices and points that will be deselected are colored in red
- Vertices and points that are currently selected are colored in green

Vertex Selection

This method selects the actual vertices on the mesh. As you hover over the selected mesh, different vertices will highlight, pressing the Vertex Select Key will select this vertex.

Point Selection

This method allows you to select not just vertices, but any point on the mesh. Sometimes it can be very useful to be able to select arbitrary points on the mesh so the colliders only go up to certain areas where vertices aren't located. Point selection is done using the Point Select Key.

Note that with point selection, it is extremely difficult to remove the point once selected, as the location is very precise. If you accidentally select a point, the best way to deselect it is using your undo shortcut.

Box Selection

With vertex selection enabled, clicking and dragging in the scene will start a box selection. This allows you to select, or deselect lots of vertices all at once. The colors shown over the vertices represent the same things they do in normal vertex selection.

Additionally, holding down while dragging allows you to only select (Box Select+ Key in preferences) or only deselect (Box Select- Key in preferences) vertices within the box.

Note that if the keycodes are set to an EventModifiers key (https://docs.unity3d.com/ScriptReference/EventModifiers.html) then the keycode must be pressed after starting a box selection, otherwise it will not work. This allows other shortcuts to work, like holding alt and left click dragging to rotate around a gameobject.

Grow Selected Vertices

Once at least one vertex is selected, this button appears which allows you to grow the selected vertices outwards from the currently selected ones. This button expands outwards along edges for all currently selected vertices.

Grow Last Selected Vertices

This button functions similar to the Grow Selected Vertices button. The difference is that this button only grows from the vertex, or vertices that were selected with the last operation.

Ring Select

This button appears once at least 2 vertices are selected. It attempts to follow triangle edges around an object, creating a ring of vertices. Once it reaches one of the vertices it has already selected, it ends.

Deselect All Vertices

This button allows you to quickly deselect all currently selected vertices in case you made a mistake, and don't want to go through multiple undo operations.

Invert Selected Vertices

This button deselects all currently selected vertices, and selects all currently unselected vertices. This can be helpful if there’s only a few vertices you wish to not include in the collider generation.

Collider Creation

Capsules, and spheres, both have multiple algorithms available to them. The method that is set by default generally works the best in most cases. If this method does not generate what you are looking for, it is easy to undo, change the method, and recalculate the collider with the same selected vertices.

If this still does not generate an appropriate collider, double check that only the points you want are selected, or add / remove vertices.

Be sure to check out the examples section for demonstrations of point selection.

Boxes

- Select all the vertices in whichever order you wish for box colliders, and they will all be included within the box.

Capsules

- Best Fit Method: The first 2 points selected define the height of the inner cylinder portion of the capsule collider. The other vertices are used to calculate the radius of the capsule. The easiest way to use this method is to select the first 2 points for the proper height, then use the ring select method on another 2 vertices to create the radius.

- Min Max Methods: The order of selection of points with min-max methods is not important. The radius and diameter methods add additional height (based on radius, or diameter) to the capsule collider during generation.

Spheres

- For all sphere methods, select points in whatever order you wish.

- The Best Fit method calculates a sphere where the points selected most closely fit the surface of a sphere. This is most useful when you have a surface where there is a portion of a sphere on the surface of a mesh. Selecting more points generally makes this algorithm more accurate.

- The Distance method can be slow with large amounts of points selected. In these cases it falls back to a less accurate calculation. The fallback calculation will not be used unless an extreme amount of points are selected.

Rotated Colliders

- Rotated box colliders are aligned along an axis defined by the first 3 points that are selected. The easiest way to select appropriate points is to select 2 vertices along a long straight edge of a triangle that aligns with the collider you want. Ideally, the third point should be the other point of this triangle. If this is not possible, try to visualize a line that connects the first two points. This line should align with the collider you are creating. The third point would lie along a flat plane, or one side of a box, that also aligns with the rotation of the collider. See the examples for more details.

- Rotated capsule colliders are aligned based on the first two points selected. The third point is included, but does not really matter. This is because as you rotate a capsule collider around it's height axis, the world-space area it covers does not change like a box collider.

- If the Attach Collider To field is a different object than the Selected GameObject, you may not need to use rotated colliders. If the gameobject you're attaching to has an axis that aligns with the direction of the collider you wish to create, you can create regular colliders. This is likely the case when creating a collider on skinned meshes. If the bones are aligned correctly with the geometry, regular colliders can be attached to the bone itself.

Mesh Colliders

- Mesh Colliders require at least 4 points to be selected. Additionally, all 4 points must not be collinear (lay along a single straight line) or helpful errors will be displayed.

- Points can be selected in any quantity greater than 4, and in any order that you wish. All points selected will be used to create the mesh used in the mesh collider. The more points you select, the more complex the resulting mesh collider will be. It is better to select as few vertices as possible that still allow you to maintain the desired shape.

- Keep in mind that the mesh created for use in the convex mesh collider is not appropriate to use as a non-convex mesh collider. Use the original mesh if you do not wish to use a convex mesh collider.

- This method will create a new mesh and save it at the location specified in preferences. It will try to find the Selected GameObject's path if that option is enabled, and fall back to the specified save folder if it is not enabled or the path could not be found. If neither path exists, it will save the mesh in the same folder as EasyColliderWindow.cs.

Collider Creation - Examples

The following examples should be used as a demonstration on how to select points to generate the colliders you are looking to achieve. An important part to remember is that a small change in vertices selected can have a large impact on the resulting collider. If the collider created does not match the vertices like you expected, hopefully these examples will better explain why this is the case. The improved support for undo/redo allows you to generate a collider, undo and select more, or different points very quickly.

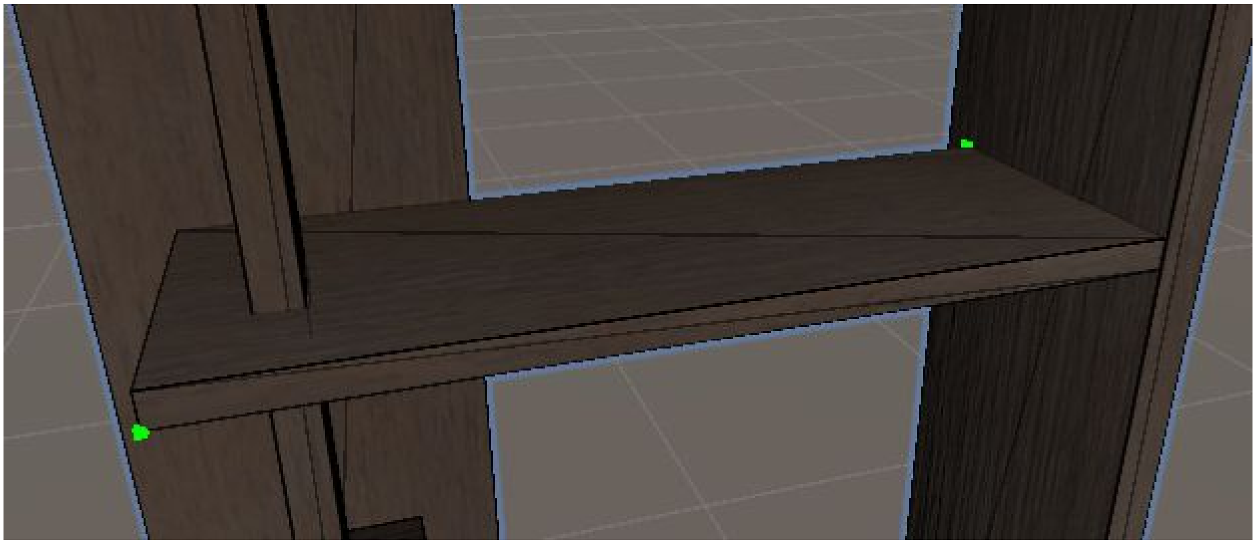
In the examples, I have set Unity's Editor shading mode to "Shaded Wireframe" to make it easier to see the mesh and it's vertices.

Note that even though some examples use static meshes, and others use skinned meshes, the concepts in each example apply to both. Skinned meshes are also used in several examples to show how you don't always need to use a rotated collider if the joint you are attaching the collider to is properly oriented.

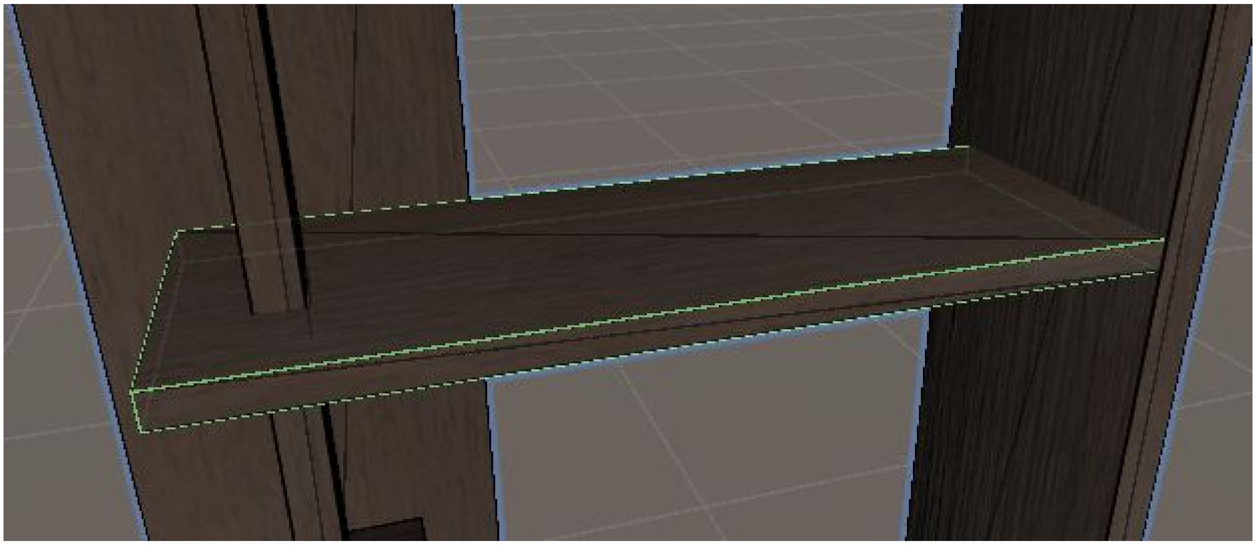
If you have any thoughts on how to make the following examples more clear please contact me at pmurph.software@gmail.com

Boxes

Example #1: Box on a Static Mesh



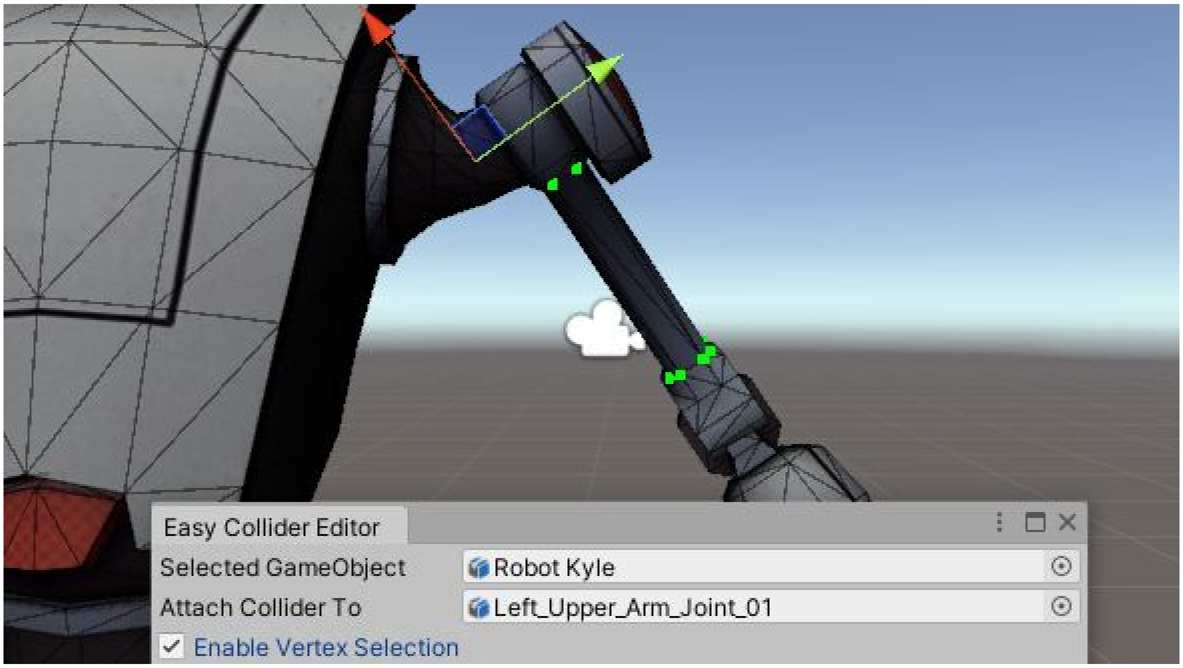
In this example we are creating a box on a simple static mesh of a shelf. We have only selected two vertices; the corners of the box we wish to create. Note that we could select as many points as we wish, and the box collider generated would contain all of these points.



After clicking the Create Box Collider button, a box collider is created that contains all of the points selected.

Example #2: Box on a Skinned Mesh

In this example we're going to add box colliders to the upper arm arm of a skinned mesh. In this case we will use Unity's free Space Robot Kyle asset.



Easy Collider Editor

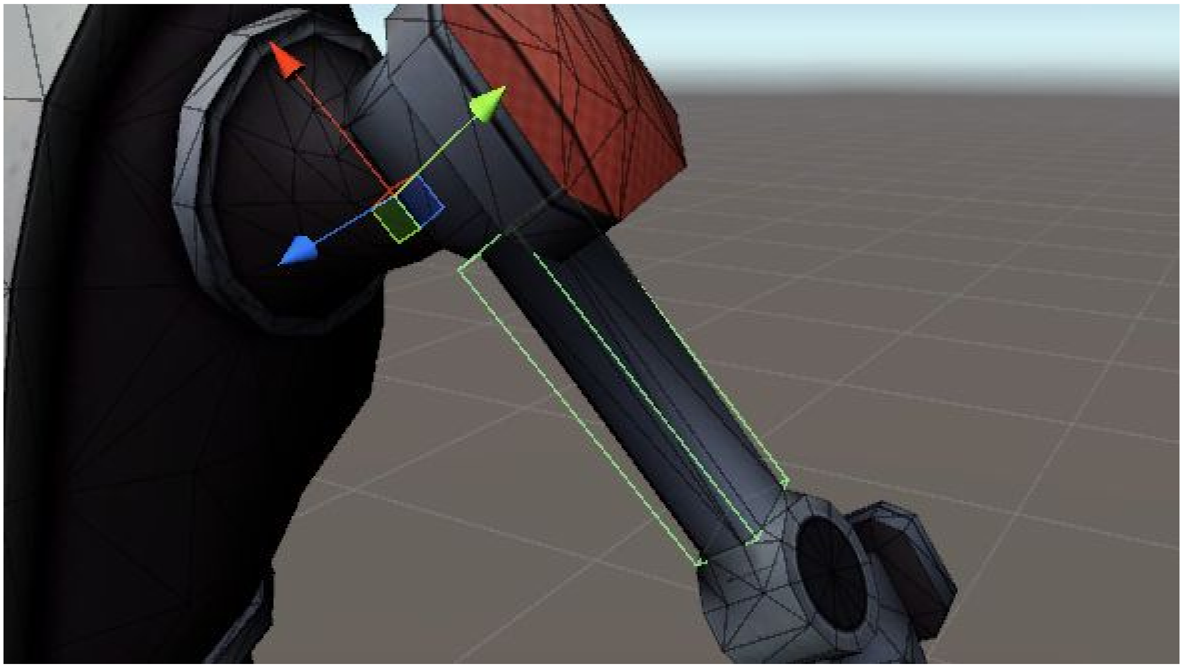
Selected GameObject: Robot Kyle

Attach Collider To: Left_Upper_Arm_Joint_01

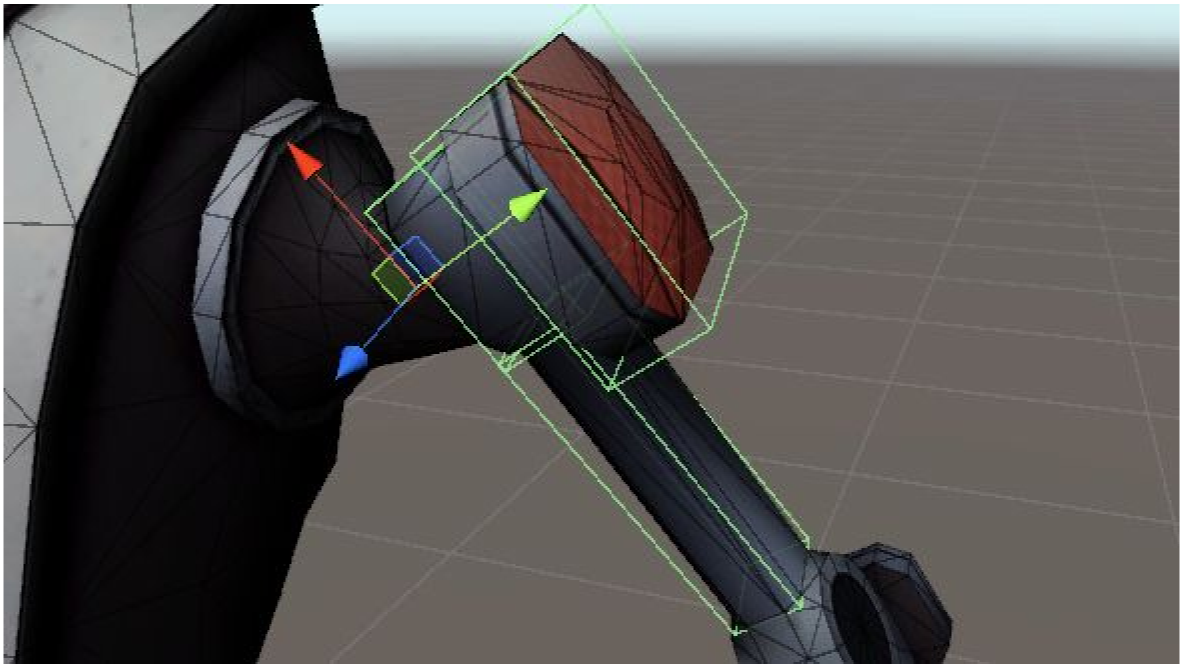
☑ Enable Vertex Selection

We have selected the root object of the whole mesh, Robot Kyle, while making sure that the Include Child Meshes toggle is checked. The Attach Collider To field was changed to the Left_Upper_Arm_Joint_01. Notice how the axis of this joint aligns with the boxes we wish to create. In properly created skinned meshes, a good alignment is usually the case. This means we can use regular colliders instead of rotated colliders.

In this image we have also selected some vertices on the skinned mesh to create a box collider from.



This box collider was created with the Create Box Collider button. Notice how it is aligned with the bone and encapsulates all the points we selected.

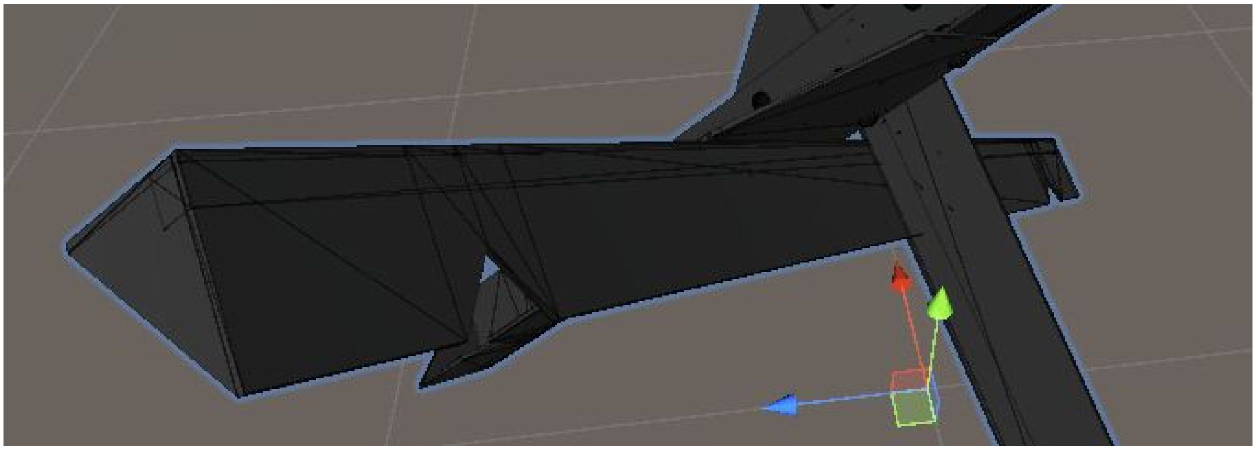


After creating 2 more box colliders, we have created very appropriate box colliders for the left upper arm of Robot Kyle.

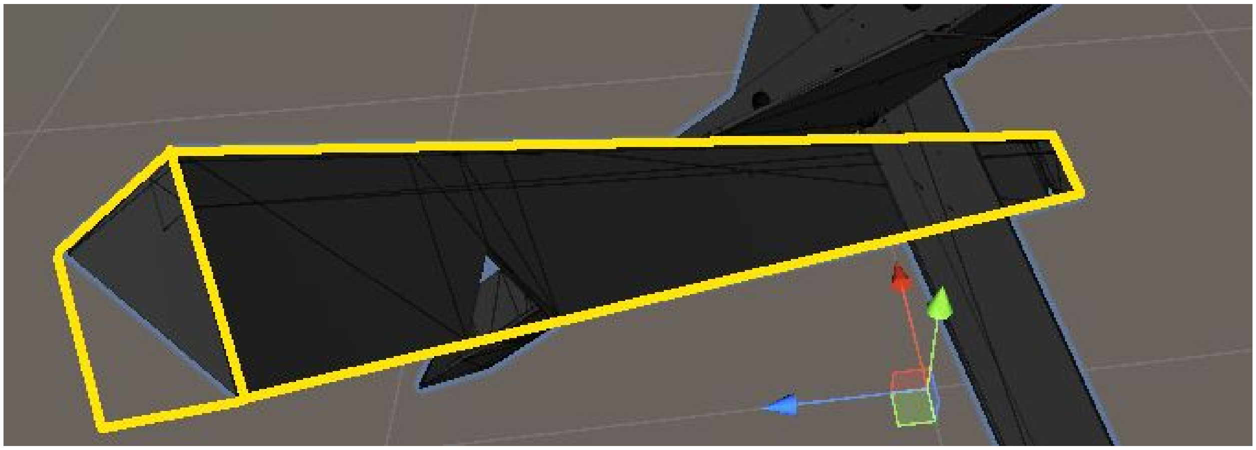
Note that if the joint was not aligned correctly we would have to use rotated colliders. We would still want to set the Attach Collider To field to the same joint as the properly aligned one, as this transform rotates with the arm during animations, which would rotate the colliders attached to it in the same way.

Rotated Boxes

Example #1: Rotated Box on a Static Mesh

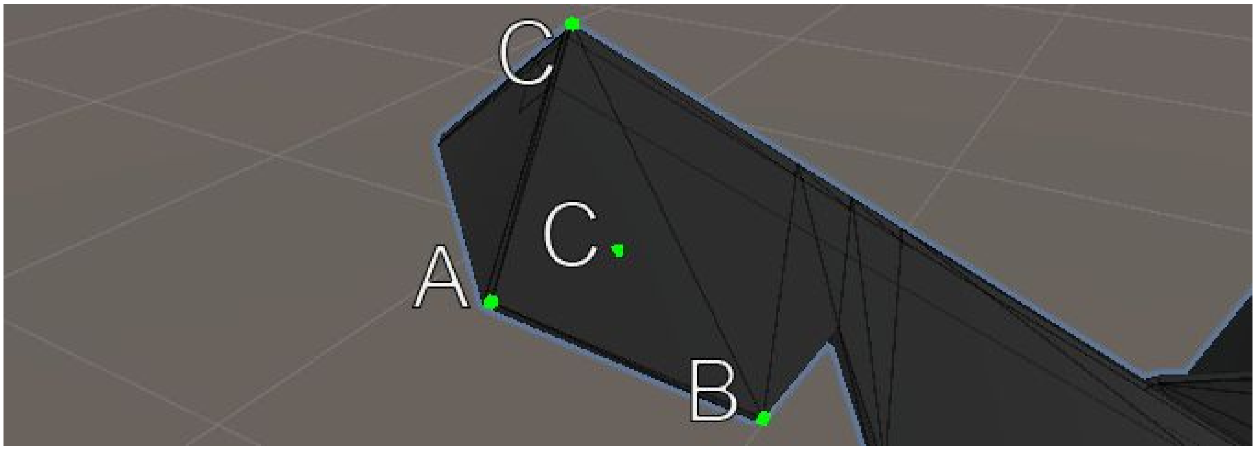


This is the object we are going to create a box collider on. Notice that it’s pivot is aligned with the world space axis’, but the geometry is not aligned with any particular axis. This makes it a good candidate for a rotated collider. In this case, I am looking to create a box collider that contains the whole angled section of the object.



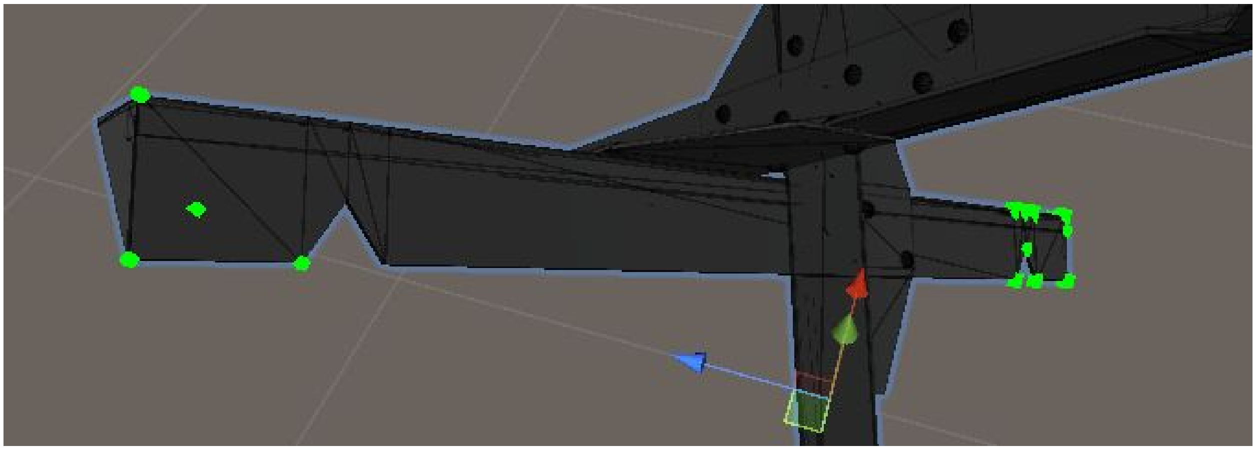
In this image I have roughly outlined the rotated box collider I am looking to make. In this example I want to create a single box collider that contains the whole angled portion

If I wanted, I could create several colliders, one for each side for more accuracy. In this case I would only need to create one rotated box collider. For the other colliders I could change the Attach Collider To field to the first rotated box collider that was created, and create the other colliders as regular box colliders.

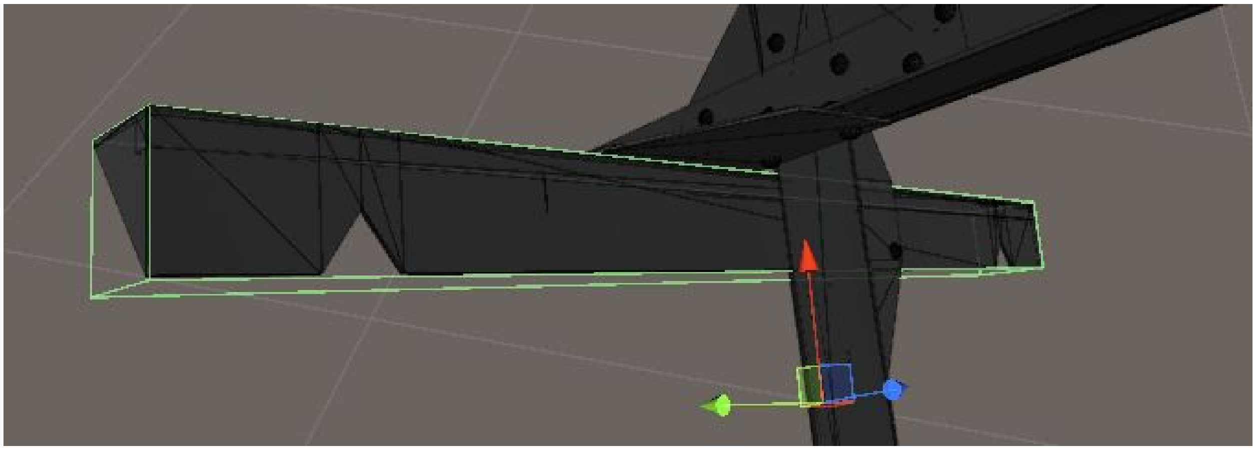


This shows the key points selected for a rotated box collider. A and B are the first two selected vertices. Notice how these two vertices align in the direction of the box collider we want to create. Note that A and B could be selected in either order.

Although there are two C's in the image above, either individual one would give the same result. Notice how when combined with A and B, either C would result in a face that would align with the resulting box collider.



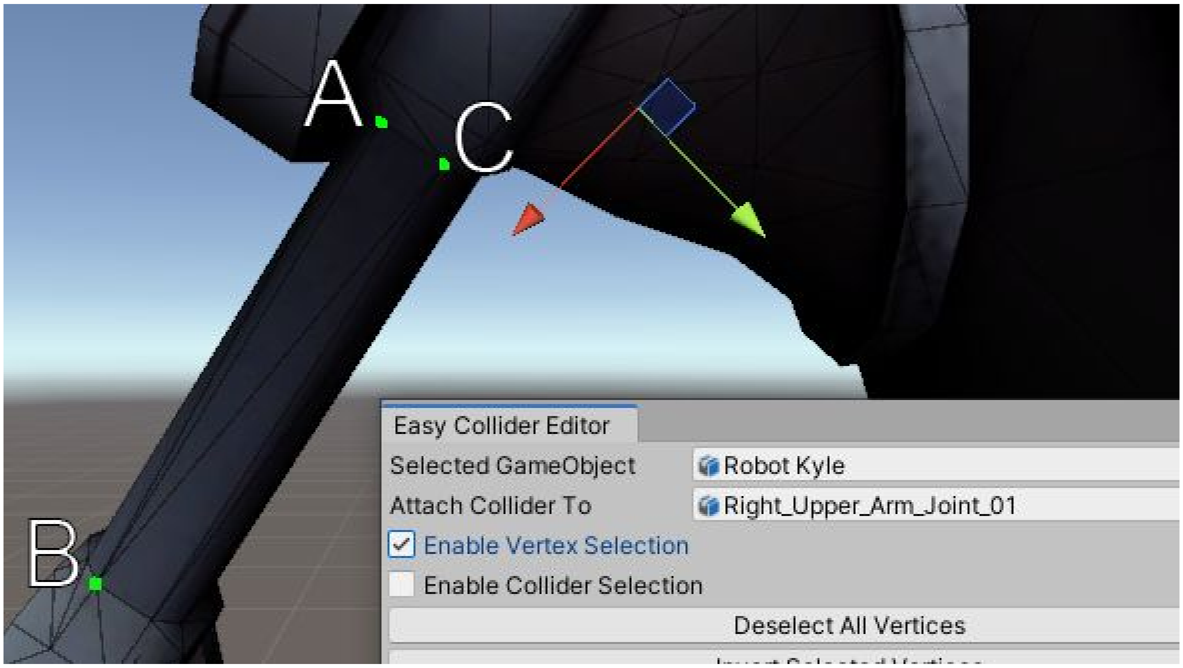
The vertices at the other end were selected very quickly by clicking and dragging the mouse to do a box selection. They are all inside the box that I want to create.



The resulting rotated box collider from the points selected above.

Example #2: Rotated Box on a Skinned Mesh

In this example, we are going to use the Right Upper Arm of Unity's free Space Robot Kyle asset.



Easy Collider Editor

Selected GameObject: Robot Kyle

Attach Collider To: Right_Upper_Arm_Joint_01

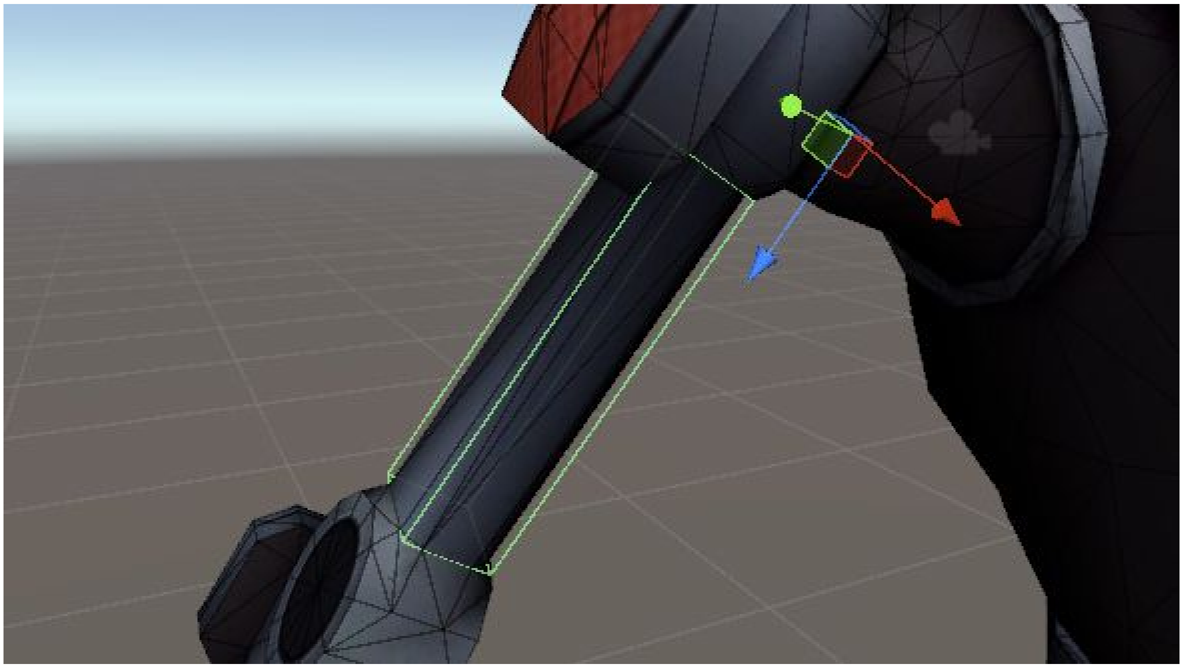
☑ Enable Vertex Selection

□ Enable Collider Selection

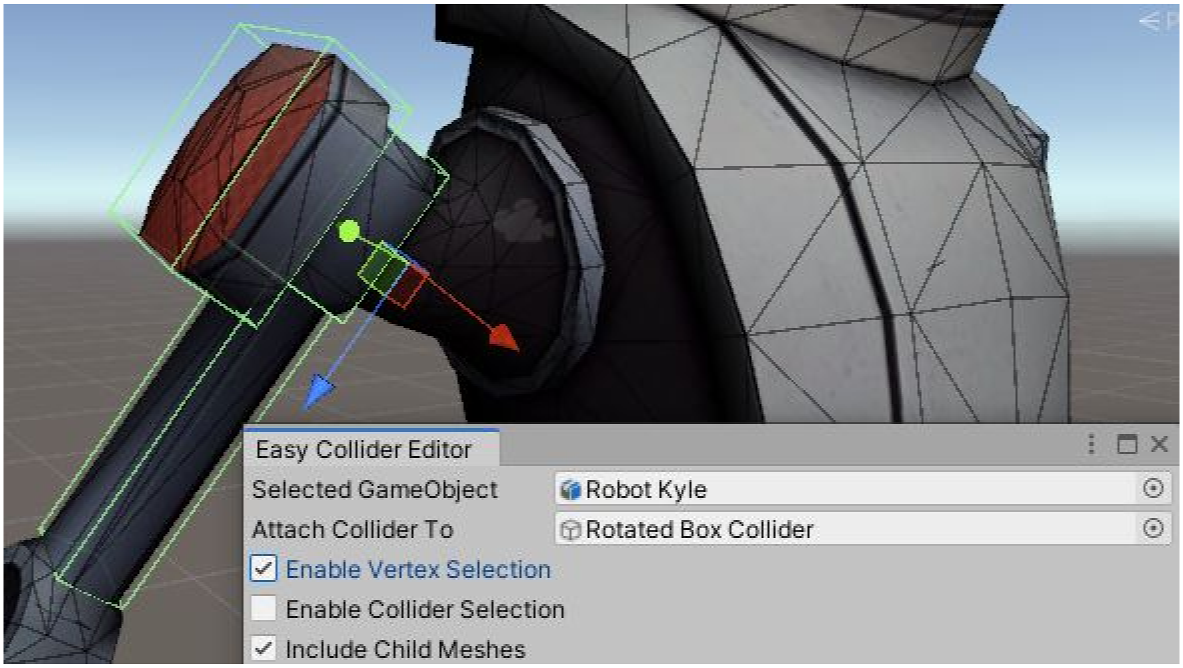
We have selected the root of the skinned mesh, Robot Kyle. As you can see in the image above, the box collider we want to create does not properly align with the arm's axis. This is a case where a rotated box collider would be needed.

Note that even though we are going to create a rotated box collider so it aligns properly, the Attach Collider To field was changed to the misaligned arm joint. This is needed because that is the joint that rotates when animations are playing, so we need to attach our colliders to that bone.

In the image above, you can also see the three points selected to align the box we want to create. In this example the order they were selected was A, B, then C. The points selected in any of the following orders: A, B, C, or B, A, C, or A, C, B or C, A,B would all create a properly aligned box in this case. Selecting B, C (or C, B), then A, would create a box that is misaligned.



After selecting a few more vertices to create the size of the box, we clicked the Create Rotated Box Collider button. Notice how the box collider is aligned with the vertices we selected above.



Easy Collider Editor

Selected GameObject: Robot Kyle

Attach Collider To: Rotated Box Collider

☑ Enable Vertex Selection

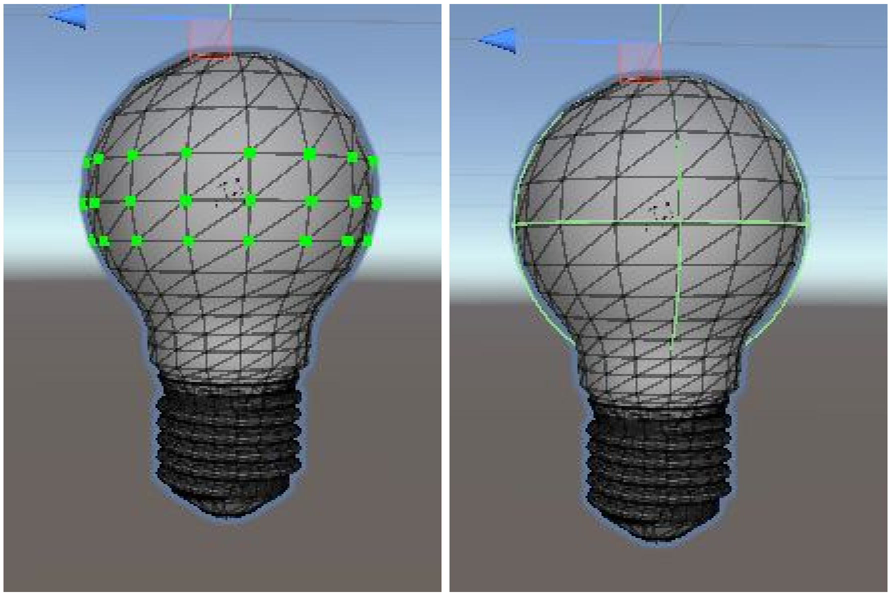
□ Enable Collider Selection

☑ Include Child Meshes

Now that we've created one rotated collider for this joint, we can now use regular box colliders. Since the Rotated Box Collider object that contains the first rotated box collider is aligned properly, it can be used in the Attach Collider To field. After this is done, we have created two additional normal boxes by selecting vertices and clicking the Create Box Collider button.

Spheres

Example #1: Min Max on a Static Mesh

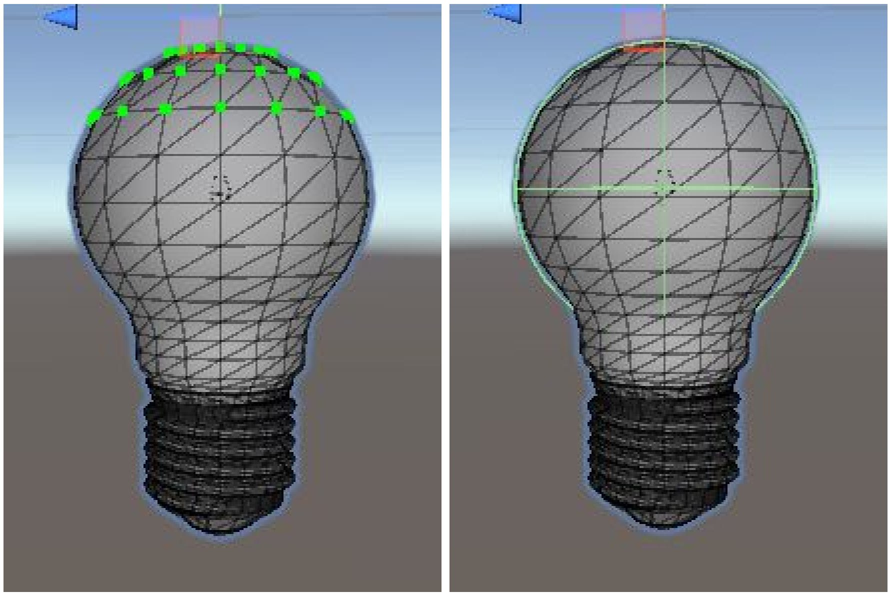


In this example the Sphere Method is set to Min Max. The vertices were selected around the center of the sphere using a ring selection. The result of creating the sphere collider is seen in the image on the right.

In this example, selecting all the vertices with a box selection results in a sphere that is too large. This is because it is not a perfect sphere. Additionally, it is difficult to guess where the bottom of the sphere would be, given the shape of the mesh changes into more of a cylinder. This is an example where changing the vertices selected changed the resulting sphere collider significantly.

Example #2: Best Fit on a Static Mesh

This example uses the Best Fit method. As you can see in the image below, only some vertices of the sphere were selected. This was done by selecting the top vertex and using the Grow Selected Vertices button.

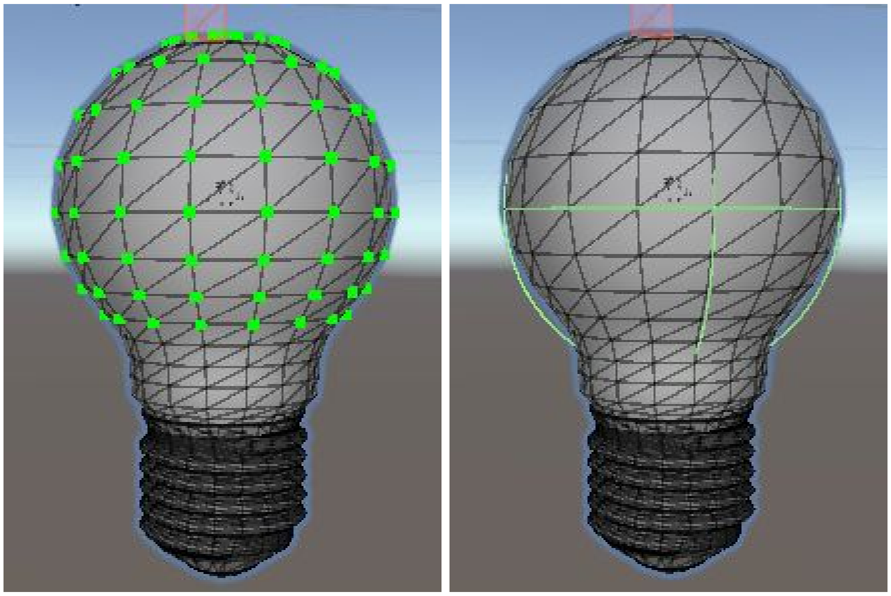


In the case of this lightbulb mesh, it was important I did not do a full box select and try the best fit method. With this lightbulb mesh, there is actually a filament inside with additional vertices. Selecting these vertices would have included them in calculating the sphere, and significantly altered the result.

As you can see, the vertices selected all lay on the surface of the sphere collider I want to create. This method calculates a sphere that has a surface that most closely matches the selected vertices. Generally adding more vertices that lay on this surface increases the accuracy of this calculation.

Example #3 Distance on a Static Mesh

In this example we have changed the Sphere Method to Distance. On the left we have used a simple box select to select all the vertices we want to use to calculate the sphere. The result is seen in the left image.

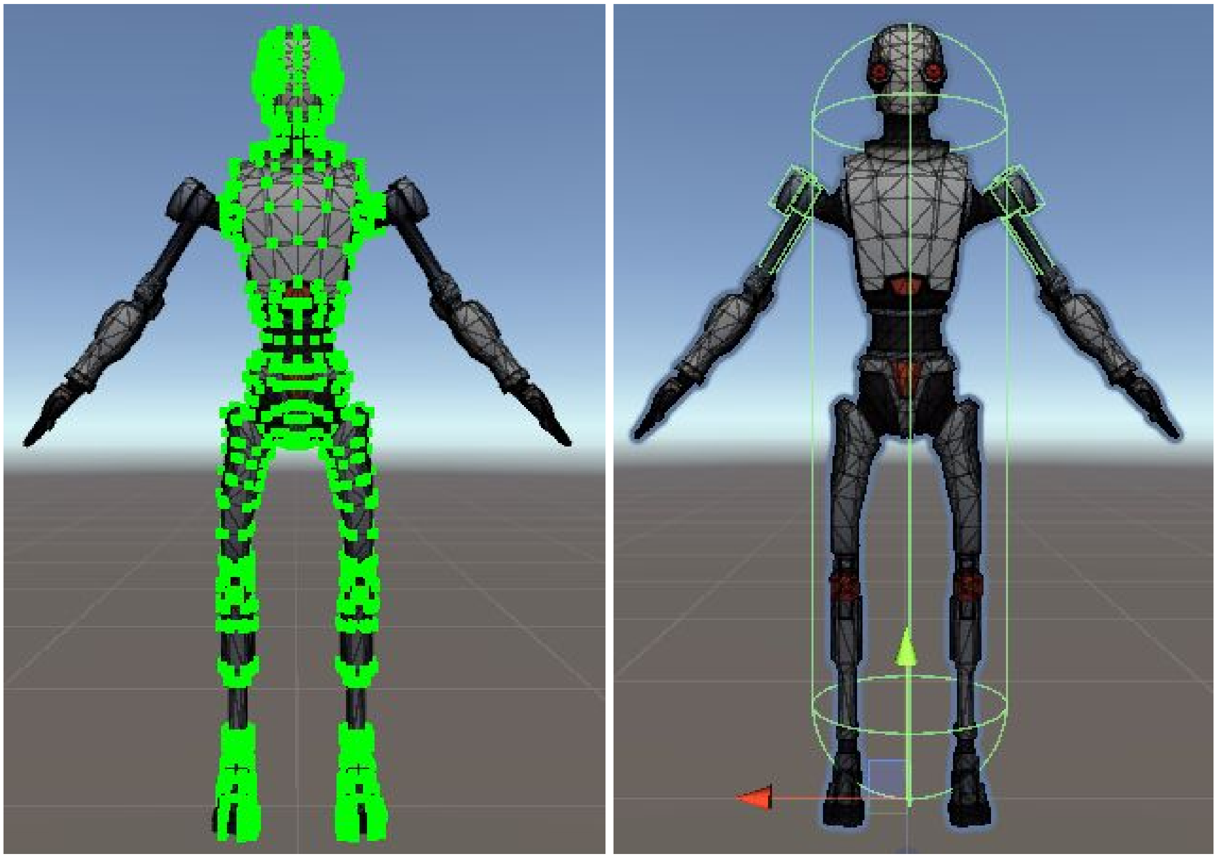


If this mesh had a very large amount of vertices, and they were all selected, this method would fall back to a less accurate but faster method which uses the average position of each vertex. In this case the distance method works pretty well as the furthest two points are on opposite ends of the sphere we want to create.

Capsules

Example #1: Min Max on a Skinned Mesh

Again we are going to use our friend Space Robot Kyle in this example. In this example we used a box select to quickly select a bunch of vertices on the characters mesh. Then, we made sure the Capsule Method property was set to Min Max. After clicking the Create Capsule Collider button, the resulting capsule collider is seen in the image on the right.

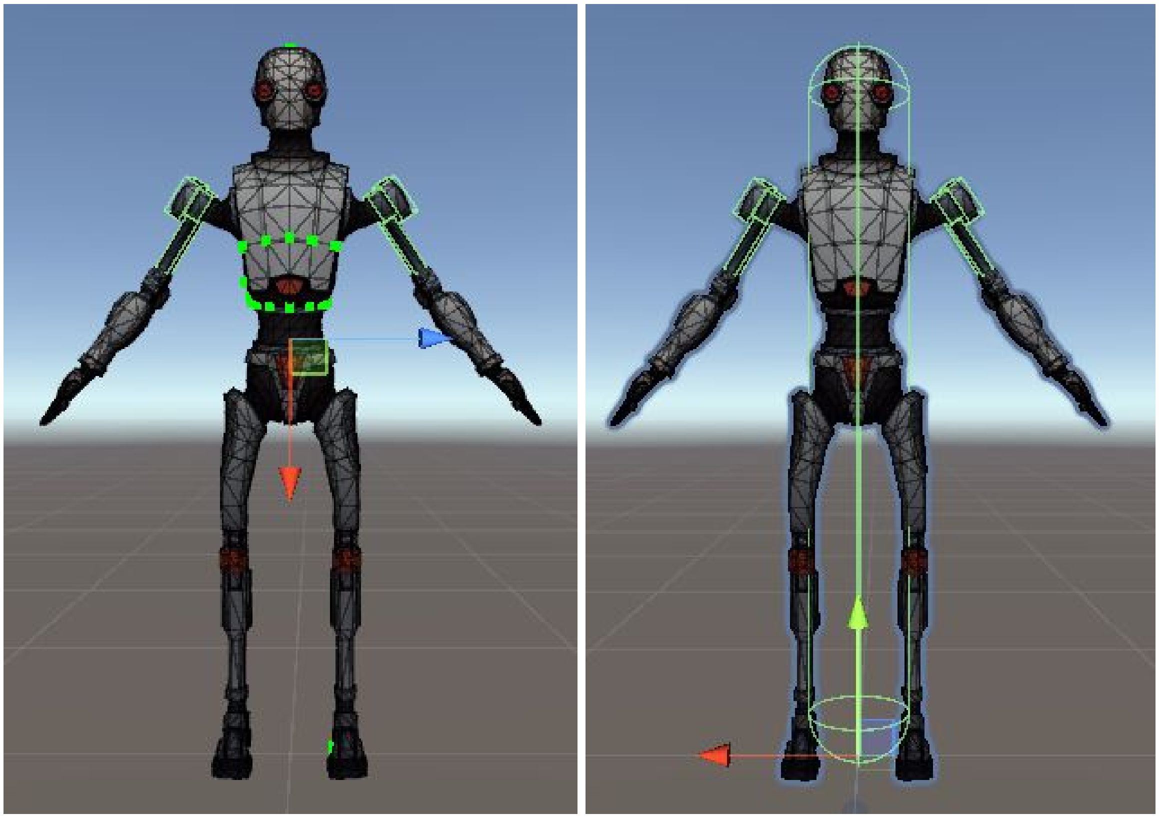


If you look at the generated collider you will notice that it does not contain all of the points that we selected. This is only the case at the top and bottom of the capsule collider, especially noticeable at the feet. In most cases this is actually what we want from a capsule collider.

Methods to add additional height are also available through the Min Max Plus Radius method, and the Min Max Plus Diameter method.

Example #2: Best Fit on a Skinned Mesh

In this example, we changed the Capsule Method field to Best Fit. In the Best Fit method, the first two points define the height of the capsule collider. In this case we selected a point on the foot, and one at the top of the head as seen in the image on the left.

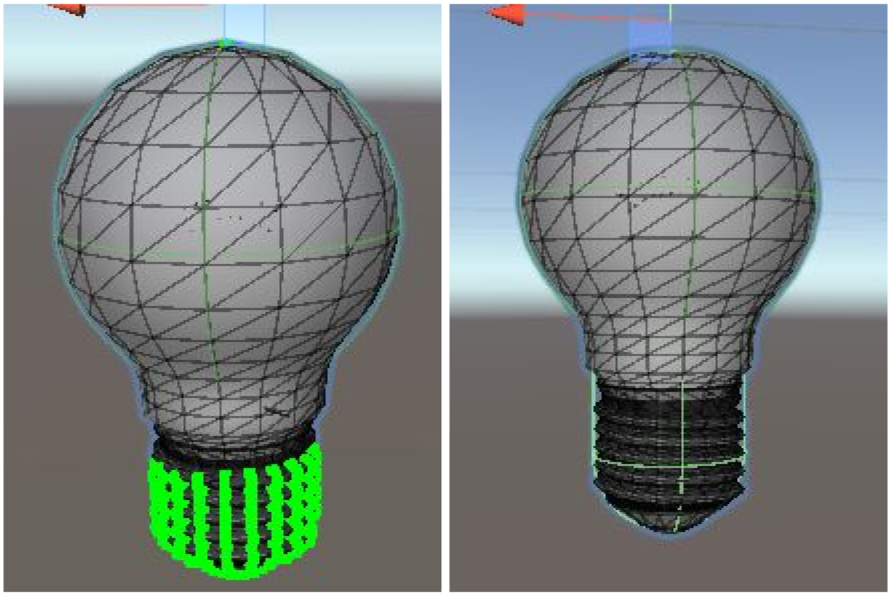


After this we selected two points in the mid section of the character, and used the Ring Select button to quickly get a set of vertices that would roughly lay on a sphere with a radius that matches the collider we want to create. The image on the right is the result of creating a capsule collider with these points selected.

Note that after the first 2 points, the best fit capsule method uses the same calculation as the best fit sphere method. The first 2 points define the height of the capsule, while the remaining points are used to calculate the sphere that would best fit those points.

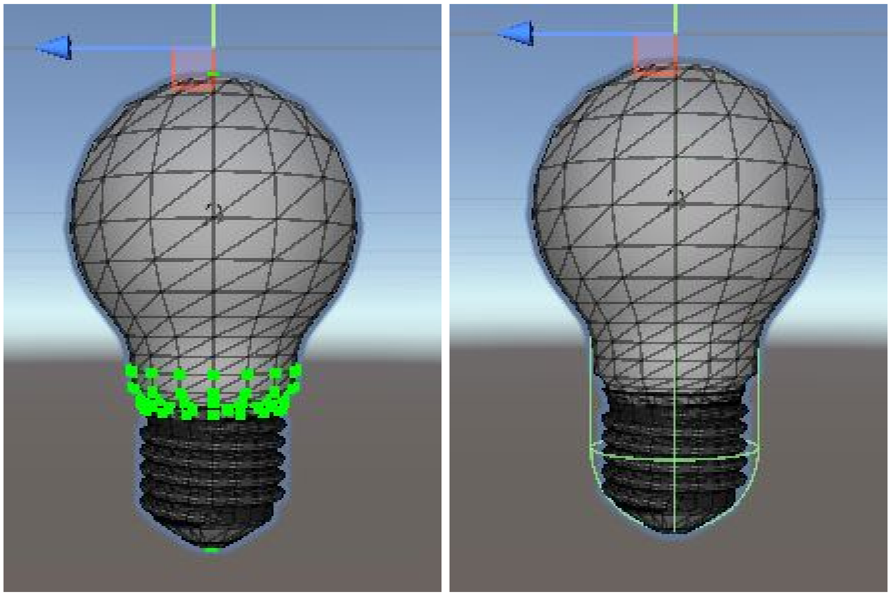
Example #3: Min Max on a Static Mesh

In this example we are looking to add a capsule collider to our lightbulb using the Min Max method. A box select was used to select all the points at the bottom of the lightbulb, and a single vertex was selected at the top of the bulb to make sure the capsule extends all the way up to it.



The result of creating the capsule collider is seen on the right. It’s hard to see in the image, but the capsule extends all the way up to the top portion of the lightbulb.

Example #4: Best Fit on a Static Mesh



In this example we are trying to add a capsule collider to our lightbulb using the Best Fit method. The first two vertices selected to define the height of the capsule, were chosen to be the very bottom of the bulb, and the very top of the bulb.

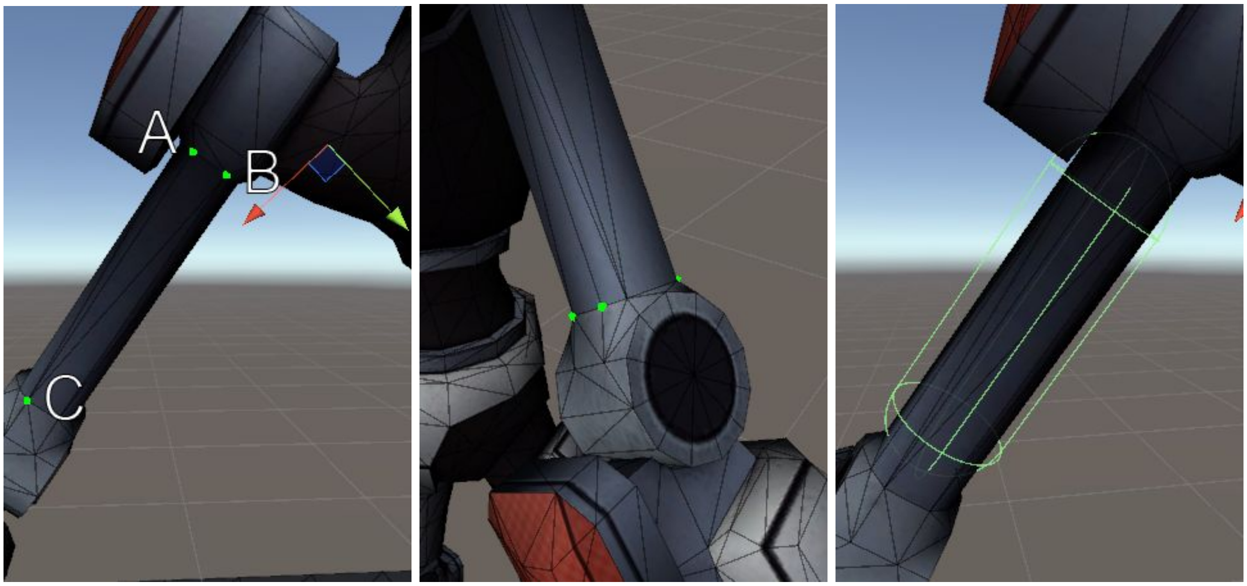
The other vertices selected lay on a sphere that has a radius similar to what we want our capsule collider to have. If we have selected a single ring around the bulb in any location, this would throw off the calculation significantly. This is because a single ring can be on a surface of a sphere of any radius equal to or larger than the ring itself. By selecting multiple rings, it allows the algorithm to better calculate the radius of a sphere that would best fit the points selected.

Rotated Capsules

Example #1: Min Max on a Skinned Mesh

Again we're going to use our friend Robot Kyle. In this case we are again using the same right arm joint used in the rotated box collider example, except this time we're doing a rotated capsule. In this example we're also using the Min Max Capsule Method.

The idea of creating a rotated capsule is similar to that of creating a rotated box. The first 3 points define the rotation of the collider. In this case, to show how a different selection order would still work, the points were selected in the order A, B, then C in the image on the left below.

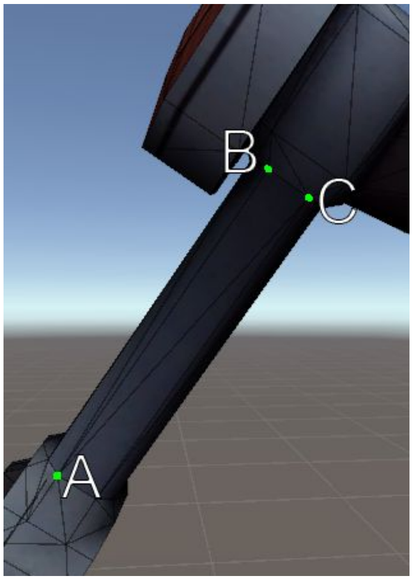


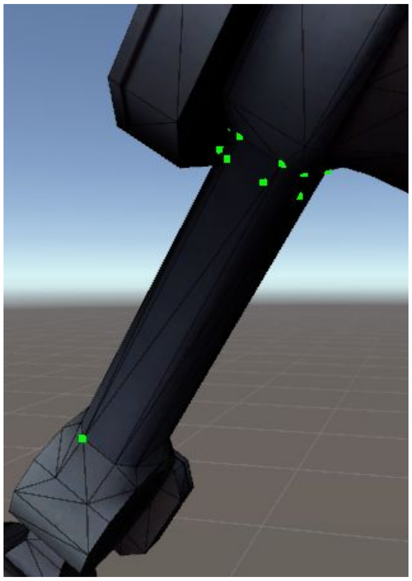
After selecting the three points, a couple more points were selected around the bottom portion of the arm. Although we could have used points around the top as well. The result of creating a rotated capsule collider using the Min Max Method with these vertices selected is shown on the right.

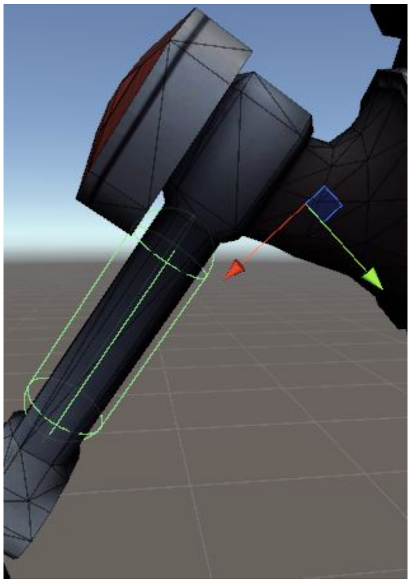
In both the Min Max method and the Best Fit method, the first three points are also included in the other calculations similar to the rotated box collider. This is something to keep in mind.

Example #2: Best Fit on a Skinned Mesh

Below you will see the same example arm, except this time the Best Fit Method was used.







The vertices on the left were selected in the order A, B, and C. The first two points, A and B, are used to calculate the rotation, and the height of the capsule. Since a capsule as it rotates around the height axis, doesn't actually change it's dimensions like a box collider does, the placement of point C is not all that important for defining the rotation.

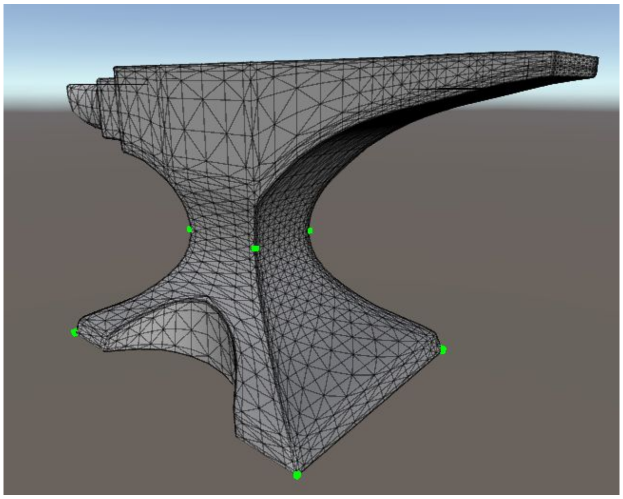
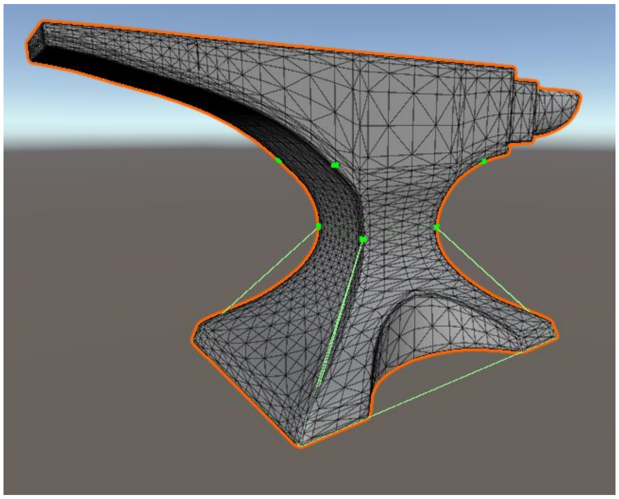
The image in the middle demonstrates how points around the upper arm joint were selected. These were selected since combined, they all lay on the surface of a sphere with a radius we are looking for. Note that general point selection was used to select points that also lay on the surface of the mesh, but aren't actual vertices of the mesh.

The result of creating a rotated collider with these is seen in the image on the right. If you wish to use the best fit method, the key point is to remember that point A and B define the height of the capsule, as well as the rotation of the capsule.

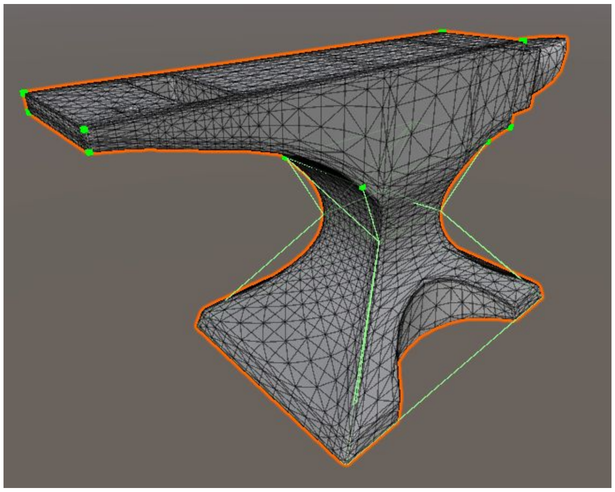
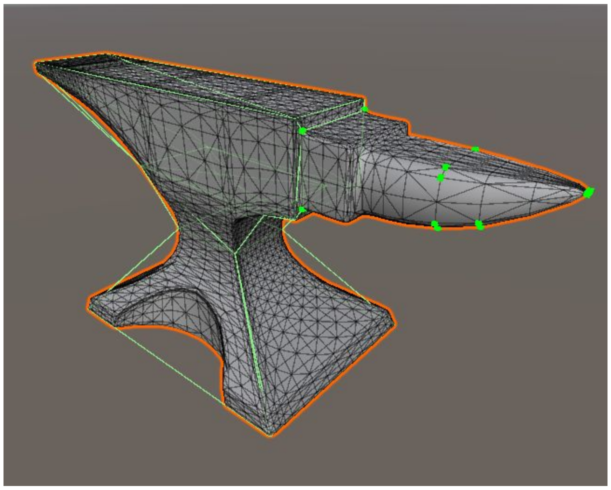
Mesh Colliders

Example #1: Static Mesh

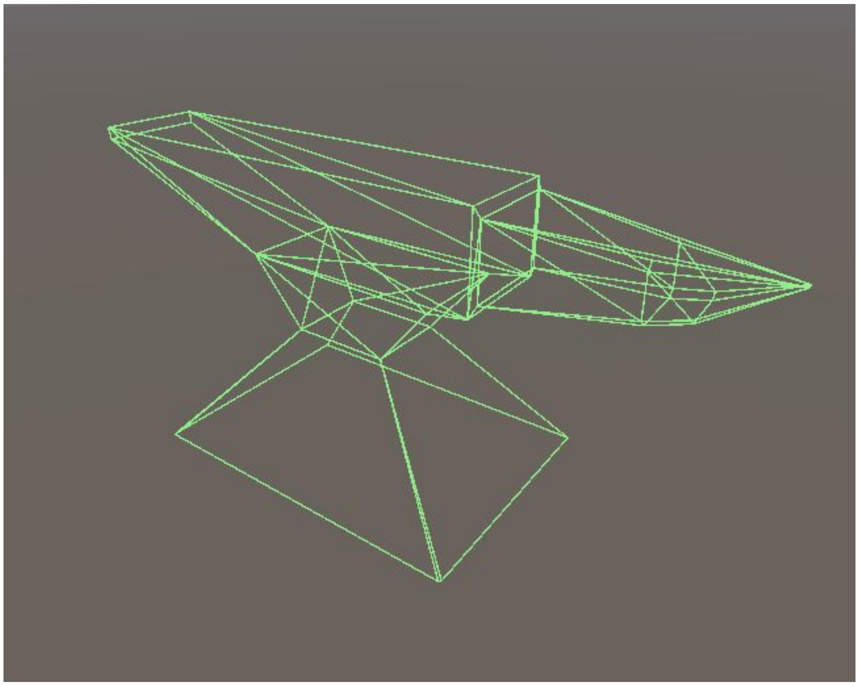
In this example we are going to create several convex mesh colliders to better approximate the shape of an anvil.

In the image on the left, 8 points were selected to create as simple a convex mesh as possible while still maintaining a rough outline of the mesh. In the image on the right you can see the result of creating a convex mesh with those points selected. Additionally, more points were selected to represent the general shape of the rest of the base of the anvil.

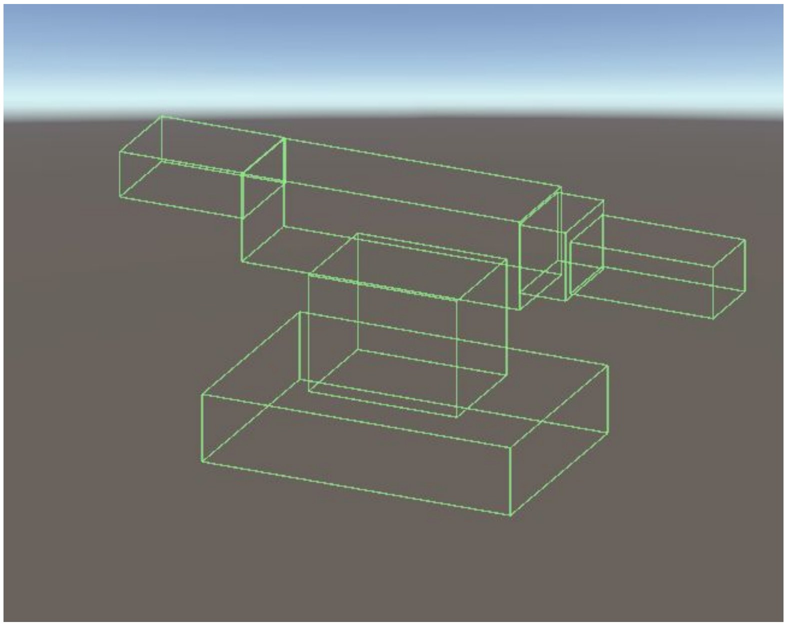



In the image on the left, the points were selected to contain the back half of the anvil. Again a low number of vertices were selected to contain the rough outline of the part of the mesh the mesh collider will contain. The image on the right followed the same process, several more points were selected along the curved surface to better approximate that section of the mesh.



Above is the final result of all of our created convex meshes. To me, this is a good approximation of the anvil mesh and will suit our purposes. If we find we want more accuracy at the base of the anvil, we could remove those two mesh colliders, and then create three mesh colliders to better represent that curved surface.

If this were not a demonstration, we should think about if we really need mesh colliders to represent a mesh of this shape. As an example, the result of using only box colliders on the anvil mesh is shown below.



Important Note on Prefab Isolation Mode

When using this asset with prefab isolation mode, be sure to enter prefab isolation mode before setting the Selected GameObject field. Once you are done editing the prefab, be sure to click the Finish Currently Selected GameObject button before exiting prefab isolation mode, otherwise components will be left on the prefab.

The correct process is as follows: Enter prefab isolation mode for the prefab you wish to edit, drag the gameobject from the hierarchy into the Selected GameObject field, create colliders, click Finish Currently Selected GameObject button, exit prefab isolation mode.

I plan to improve prefab isolation mode functionality once the requirements for me to better implement it are out of Unity's Experimental namespace and in their long term support streams.

Other

FAQ

1. Vertex selection isn't working / I cant see the vertices I have selected.
    - Make sure the Selected GameObject field is set to the gameobject you want to work with.
    - Try changing the Render Vertex Method in preferences from SHADER to GIZMOS
    - If you are using GIZMOS, make sure gizmos are enabled in the scene view.
    - Check the console for any warnings from this asset.
    - If you are selecting a parent of a mesh, make sure Include Child Meshes is enabled
    - If the child mesh is a skinned mesh, make sure Include Child Skinned Meshes is enabled in preferences.
    - Try not to add, edit, or remove components from the SelectedGameobject and its children while you are selecting vertices.
    - If you did add, editor, or remove components, click the Finish Currently Selected GameObject button, then reselect the gameobject.

2. Using gizmos causes slowdown with lots of vertices selected, why can't I use the shader?
    - The shader used to display vertices uses a compute buffer. This is only supported on some systems. More information about platforms where compute shaders work can be found here: https://docs.unity3d.com/Manual/class-ComputeShader.html

3. The rotated colliders aren’t being rotated correctly.
    - When creating the rotated box colliders, the first 3 points selected define the rotation of the box collider. For a capsule collider the first 2 points define the rotation of the collider. These points are very critical when creating rotated colliders. Look at the examples in the examples section carefully, and try to select a different set of points.

4. Prefab isolation mode left components on the gameobject.
    - Unfortunately, most events for entering and leaving prefab isolation mode are still in Unity’s experimental namespace for current LTS versions of unity. To make sure components are not left on the gameobject, be sure to click the Finish Currently Selected GameObject button before leaving prefab isolation mode.

5. I upgraded to a new version of unity and now I can’t select vertices!
    - In new versions of unity, the most likely thing to break is something I have tried to make as easy to fix as possible.
    - In EasyColliderWindow.cs search for EASY_COLLIDER_EDITOR_DELEGATES this should bring you to the OnEnable() and OnDisable() functions.
    - In these functions, you should see SceneView.duringSceneGui
    - The current description for this event in the unity documentation is “Subscribe to this event to receive a callback whenever the Scene view calls the OnGUI method.”
    - Look for a similar event description on the new unity version’s documentation here: https://docs.unity3d.com/ScriptReference/SceneView.html
    - Change the SceneView.duringSceneGui to SceneView.the_new_event_name in both OnEnable() and OnDisable()
    - If this does not fix anything, or you are uncomfortable doing these edits yourself, please contact me at pmurph.software@gmail.com

6. Unity is giving me a warning that it could not create a convex hull because it has >256 polygons and is instead using the partial hull?

- In newer versions of unity, the inflate mesh option on mesh colliders has been removed. Since this removal, if the convex hull created from a mesh has >256 vertices physX will display a warning about having too many polygons.
- Everything will still work as expected, but instead of this warning being hidden by unity, it is now displayed in the console.
- To fix this warning, consider making each convex hull simpler by selecting as few vertices as possible during collider creation.
- Additionally, multiple mesh colliders with less than 256 polygons can be created on the same object to solve this error.
- The example in this documentation of creating convex mesh colliders on a static anvil mesh may be helpful.

Bug reports

If you experience any issues or bugs while using Easy Collider Editor, please contact me at pmurph.software@gmail.com with as much information as possible. Please include the version of unity you are using (ie 2019.3.7f1). It would also be helpful to describe what you were trying to do, what you expected to happen, and what actually happened. Images are also helpful if you are able to provide them.

Feature Requests / Improvements

In this new version of Easy Collider Editor I have included many requests from users for features and improvements. Without these requests, I don't know the types of new features or general improvements people wish to see. I love hearing what users would like to have added to this asset, as well as any improvements or clarifications that can be made to this documentation. I can't guarantee all requests will be added to the asset, but if you have any features or improvements you would like to see, please contact me at pmurph.software@gmail.com