MTree Bezier Addon

Setup:

1. Import the the MTreeBezierAddon.unitypackage
2. Open Assets/.../Mtree/Scripts/MTree.cs.
3. Scrolldown unitl there is the part with `public void AddTrunk(...){`
4. Follow the guide Lines on Page 2 to 4

Add spline to Scene:

1. Create an Empty GameObject in Scene.
2. Add Mtree Bezier to the GameObject. MTree is Auto added to GameObject aswell.
3. Press the Button Activate MTree Beuier for Bezier editing, hit again for normal editing
4. MTree Updates as soon as you edit the Bezier.
5. Add with Button Add Curve new Bezier curves.

Explenation to Bezier Curves:

Modes: Support of 3 Modes = Free / Aligned / Mirrored
Red Points = Controll Points
Yellow Points = Tangents
Click on any Red / Yellow dot and a Position Handle / Gizmo appers, ready to start editing the Bezier Curve.

Explenation to Addon:

Both modifications (Bezier and Direction of branches) are Optional. MTree works as expected with and with out these modifications.
If Setup you can add MTreeBezier as component, but MTree works aswell without.
Mtree

Mtree v 2.1
Mtree Bezier Addon by Dan Wipf
Catlikecoding - Tutorials and other stuff (Bezier calculations)

setup of *public void AddTrunk(Vector3 position,...){*

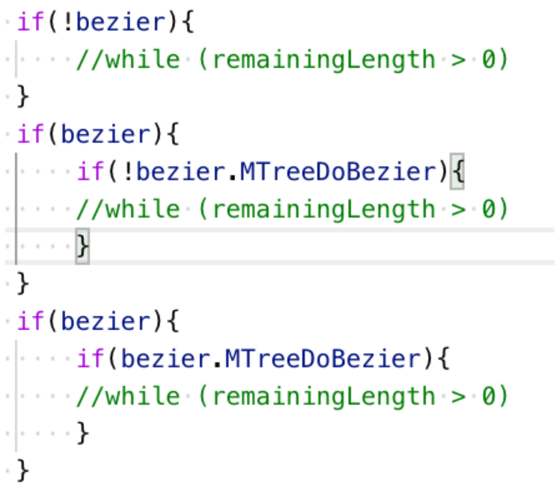
Open MTree.cs and find the Code lines public void AddTrunk(Vector3 position,...)

This will be the part which has to be edited for Bezier functionality. You will see now some simple setup code and a large while(remainingLenght > 0) part. This part we will copy later 3 times.
(Safteycheck so we do not get any errors after we are done.)

Add after the first { MtreeBezier bezier = treeTransform.GetComponent<MtreeBezier>();

Create after the stems.Add(extremity);

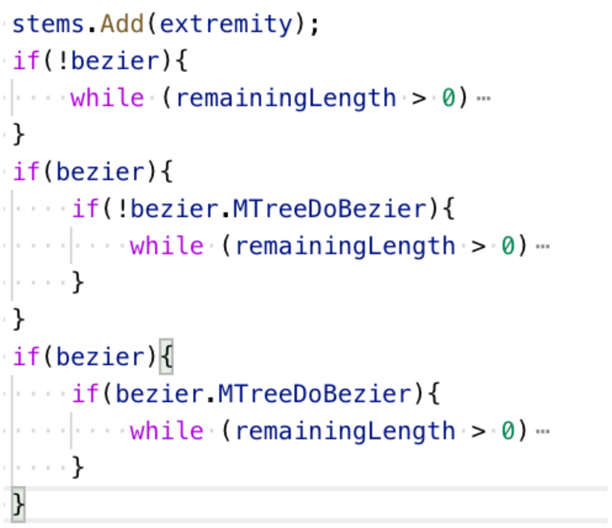
```
if(!bezier){
        //while (remainingLength > 0)
}
if(bezier){
        if(!bezier.MTreeDoBezier){
        //while (remainingLength > 0)
        }
}
if(bezier){
        if(bezier.MTreeDoBezier){
        //while (remainingLength > 0)
        }
```



```
if(!bezier){
    //while (remainingLength > 0)
}
if(bezier){
    if(!bezier.MTreeDoBezier){
    //while (remainingLength > 0)
    }
}
if(bezier){
    if(bezier.MTreeDoBezier){
    //while (remainingLength > 0)
    }
}
```

Replace each //while (remainingLength > 0) with the original while (remainingLength > 0).
now if you haven‘t delete the original while loop it‘s time to do it. now.
This hirachy of code you should have now:

```
public void AddTrunk(...){
some setup code
        if(!bezier){
                while(...){...}
        }
        if(bezier){
                if(!bezier.MTreeDoBezier){
                        while(...){...}
                }
        }
        if(bezier){
                if(bezier.MTreeDoBezier){
                        while(...){...}
                }
        }
```



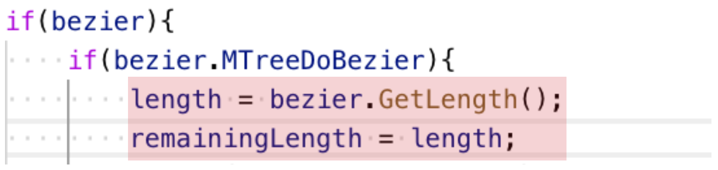
```
stems.Add(extremity);
if(!bezier){
    while (remainingLength > 0)…
}
if(bezier){
    if(!bezier.MTreeDoBezier){
        while (remainingLength > 0)…
    }
}
if(bezier){
    if(bezier.MTreeDoBezier){
        while (remainingLength > 0)…
    }
}
```

The first two if Statements we do not have to modify anymore, but the last one where the Bezier is applied we need to change a bit.

setup of *while (remainingLength > 0) loop*

1.
After the if(bezier.MTreeDoBezier) { we need to add those two lines:

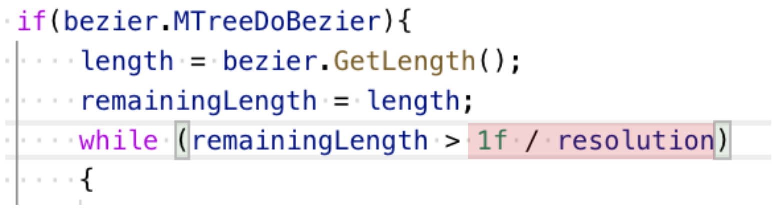
length = bezier.GetLength();
remainingLength = length;



```
if(bezier){
    if(bezier.MTreeDoBezier){
        length = bezier.GetLength();
        remainingLength = length;
```

This will override the original trunk length with the length of the Bezier.

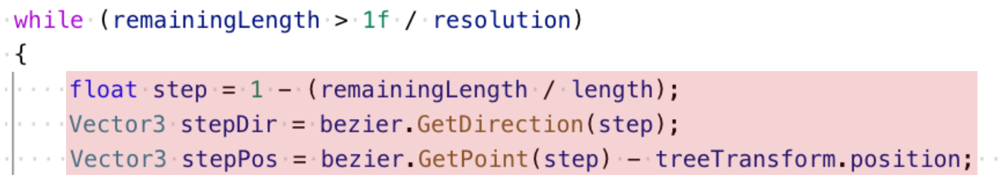
2.
Change while (remainingLength > 0) to while (remainingLength > 1f / resolution)
this will make sure that the last point of the tree is right at the end of the Bezier Curve.



```
if(bezier.MTreeDoBezier){
    length = bezier.GetLength();
    remainingLength = length;
    while (remainingLength > 1f / resolution)
    {
```

3.
Add those three lines after the while statement.

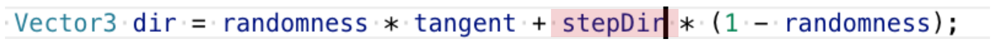
float step = 1 - (remainingLength / length);
Vector3 stepDir = bezier.GetDirection(step);
Vector3 stepPos = bezier.GetPoint(step) - treeTransform.position;



```
while (remainingLength > 1f / resolution)
{
    float step = 1 - (remainingLength / length);
    Vector3 stepDir = bezier.GetDirection(step);
    Vector3 stepPos = bezier.GetPoint(step) - treeTransform.position;
```

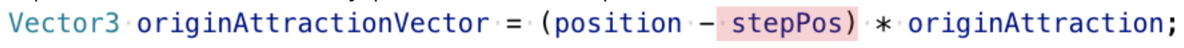
Simple setup for the current position and direction based on the current step on the Bezier Curve.

4.
Replace the one extremity.direction with stepDir

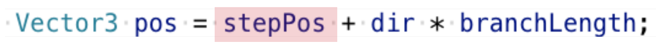


```
Vector3 dir = randomness * tangent + stepDir * (1 - randomness);
```

Replace the two extremity.position with stepPos



```
Vector3 originAttractionVector = (position - stepPos) * originAttraction;
```



```
Vector3 pos = stepPos + dir * branchLength;
```

So the position and directions of the Bezier get applied to the Tree.

So. that‘s it. Now you can start creating Bezier Trees!

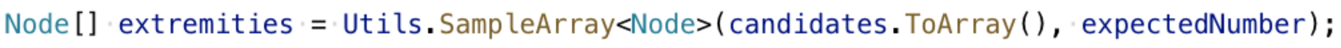
Next we setup the branch direction modifier.

- end of bezier setup -

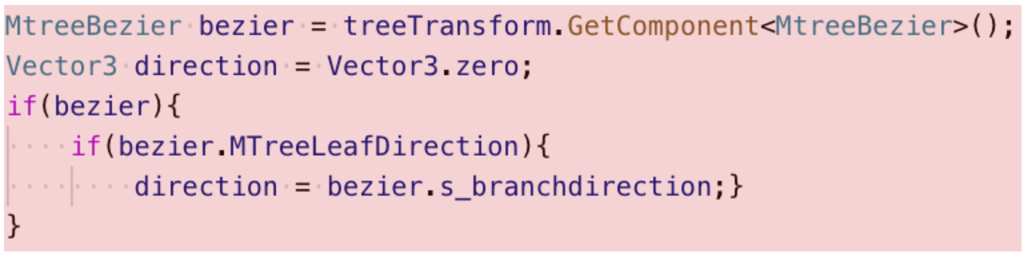
setup of *public void Split(int selection,...){*

1.
After Node[] extremities add add those lines to the Script

```
MtreeBezier bezier = treeTransform.GetComponent<MtreeBezier>();
Vector3 direction = Vector3.zero;
if(bezier){
        if(bezier.MTreeLeafDirection){
                direction = bezier.s_branchdirection;}
}
```



```
Node[] extremities = Utils.SampleArray<Node>(candidates.ToArray(), expectedNumber);
```

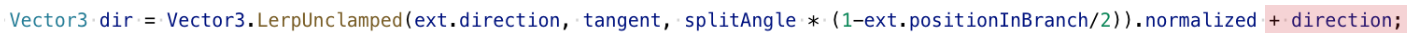


```
MtreeBezier bezier = treeTransform.GetComponent<MtreeBezier>();
Vector3 direction = Vector3.zero;
if(bezier){
    if(bezier.MTreeLeafDirection){
        direction = bezier.s_branchdirection;}
}
```

This will check for MtreeBezier and manages the offset of the Branch growth direction

2.
Add at the end of Vector3 dir = Vector3.LerpUnclamped(...);

```
+ direction;
```



```
Vector3 dir = Vector3.LerpUnclamped(ext.direction, tangent, splitAngle * (1-ext.positionInBranch/2)).normalized + direction;
```

This will add the additional direction, defined in the MTree Bezier component to the core branch direction.

- end of directional setup -