Problem solving:

Table of Content:

Performance

Summary: this block I joined a 3rd year group. Their main issue was the city. The city buildings were high poly individual meshes. Every building was more then a million polygons and the dense city contained more then 2 thousand buildings. The team wanted to keep the level of detail and have a smooth running game. With low loading time. The end result I came up with is making instances with 3 by 3 wall pieces. Calculate the position in Houdini then export in obj and convert it to a binary file. This so unreal can load it very fast.

Analysis:

The team had 3 block behind them without a result of a city. The scope of the city was bigger than the one of GTA but there is only 1 city and as dense of a city as new york. With 3 distinct districts: slums, apartments and corporate. The detail of the city should be very high. An other mechanic of the game is that you are able to fly trough the city in a fast pace since that is the main experience of the game. The layout was predefined and the mesh was triangle shaped. The buildings looked fairy similar in their district and all buildings where square shaped. The last 2 factors can work in my benefit.

Discussed solutions:

First 2 solutions that came to my mind where LOD's and instances.

With those techniques in mind. We where planning out on how to go with the shapes. 2 where discussed.

1.One was first generate the first floor. Then the second and duplicate the second one up.

2.Second was making instance on every 3 by 3 wall piece and lay them in a grid. This will created many

3.a really cheap way to create the game as well would be only boxes and place instances of other props like balcony's.

Also the combinations of 2 and 1 was discussed so make pattern in 2 floors. And then go up. However this is planned to move it after the second solution is tested

there where some problems on the way of developing the second solution. Those are discussed in other paragraphs

Tested solutions:
we started with the instancing every wall piece,

The problem was in flexibility of complex shapes but since we are using box shapes there was enough flexibility to use it, also make it into unreal was difficult. We where not able to arrange a Houdini engine license. We used the obj exporter in Houdini. And converted it in to first csv fies. However it

massively increased loading time in the game so we converted it instate to binary. As well tried a better algorithm where the points get delete who where not seen.

End result:

game runs fluid now and the loading time improved till 1 minuted. However there is still problem that some of the wall pieces seem to be rotated or displaced.

To Improve:

this time I took a big gamble if instancing would improve the performance to the right amount team for feedback. Next time I should profile smaller parts as well

Pipeline development

Summary:

the pipelines to create the city before this one where long and required many hand work hours.

Analysis:

only previews pipeline I was aware of at the start of the block. Was the one with only static objects what was developed in block c. however instancing was tried before. The static object work flow. Required to make high poly objects. And place them one by one in the scene. The other work flow of instancing required to place the walls by hand.

Discussed solutions:

One was using the Houdini engine. This required some payment from the team. Because we wanted to make a published game. However since there was already much spend money on software that not was used. The team was not prepared to pay for it.

Talked about placement in unreal itself. However this was tried before and the handling was tough. Required a lot of handwork.

What also was possible was exporting the points as objects. And later on convert them to .CSV . Read that out in unreal and make a blueprint that reads it and place instances in the game for that. And then make also geometry for corners roofs and collision.

Tested solutions with result:

We went for the last one. However the .csv loading took to long. So we went to convert them to binary. What was way faster. Also improved the game play speed. The geometry and collision was pure geometry when exported and every face had shaders. So we made a blender file that does steps. These steps where represented in a script. So only run script and export. And your done.

end result

was exporting from Houdini to obj then convert the point clouds to binary in python. And the collision and geometry get opened in blender and exported as fbx with the right structure. Before it is in game it takes 3 people. One to export in Houdini and blender. One to export to binary. And one to implement in game

to improve

The time in Houdini to cook all of it took very long. This made it take a day. One other thing that slowed it down was the fact that it has to go trough 3 people which had there own thing to do. To get it in to test phase.