CAMPUS Karte

Campus Karte

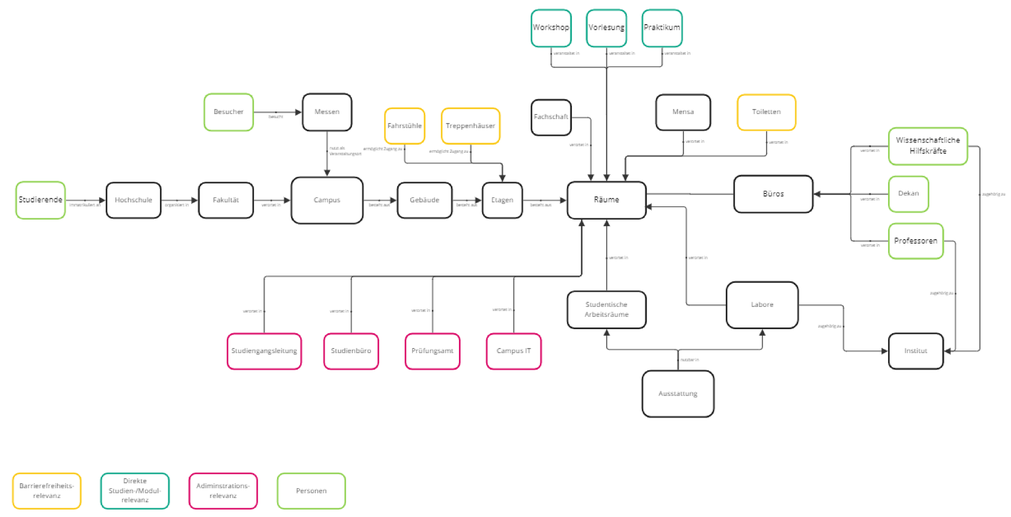
Technology Arts Sciences TH Köln

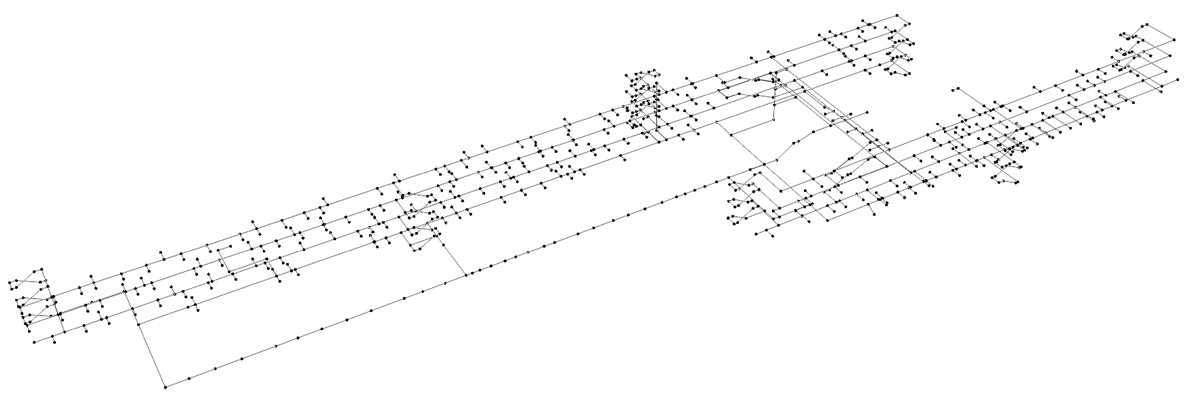
Anastasia Chouliaras
Domenic Wolf
Jens Burgdorf

Einführung

Durch die lange online Zusammenarbeit und die andauernden hybriden Veranstaltungen kennen sich viele Studenten nicht gut am Campus aus. Die Erst- und Zweitsemester haben durch die spezielle Raumnummerierung und die verschiedenen Gebäude in der Regel Orientierungsschwierigkeiten und suchen häufig lange nach dem richtigen Raum. Doch auch auf höhere Semester kann dies zutreffen. An dieser Stelle soll das Projekt ansetzen und die Orientierung durch eine 3D Karte erleichtern. Damit sollen sich Studenten, neue Mitarbeiter oder auch Besucher sowohl vor Ort als auch online ein Bild des Campus machen und die Route zum gesuchten Raum angezeigt bekommen.

Orientierung an der Hochschule





3D Kartenmodell

Das Kartenmodell wurde anhand des Grundrisses des Gebäudes auf Google Maps und einem Gebäudeplan erstellt.
Zunächst als grober Umriss, danach wurde das Model stetig iteriert und die einzelenn Räume hinzugefügt, Etagen getrennt exportiert und Auzüge und Treppen eingefügt.
Um sich auf der Karte zu bewegen wurde ein Liniennetz (siehe Links) angelegt, dass den Fluren und räumen es Gebäudes entspricht. Mit diesem Liniennetz arbeitet der Wegfindungsalgorithmus um die Route zu generieren.

Importieren des Meshs

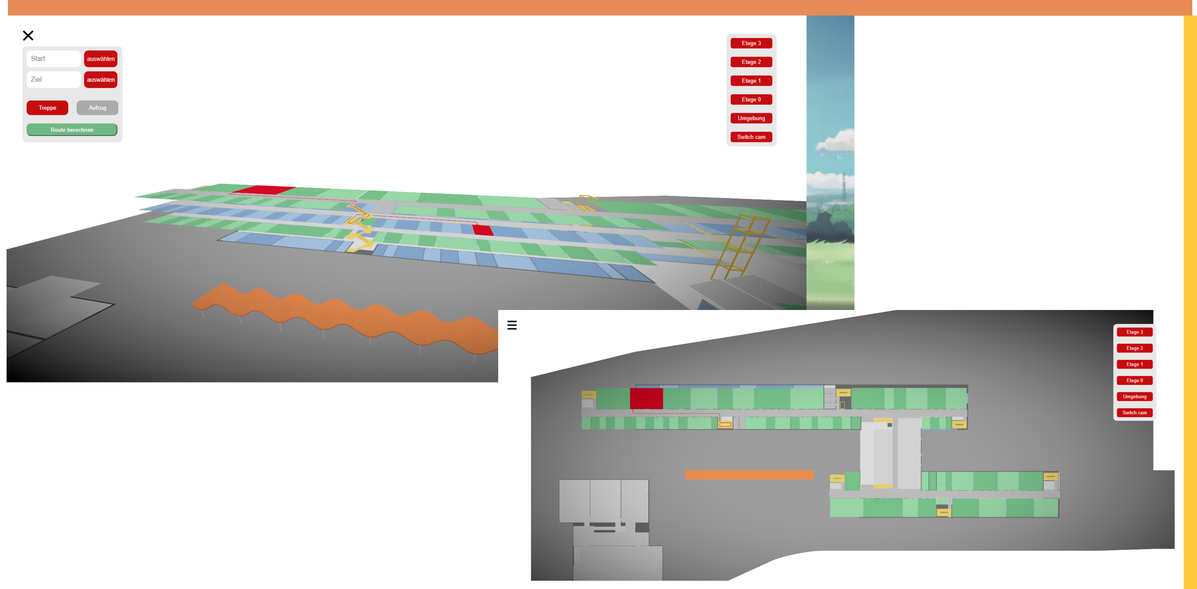
Beim Importieren des Meshes werden die einzelnen Teile gesplitet und zuerst in einzelne Punkte geschrieben die später in eine verkettete Liste aufeinander referenzieren.
Mit der funktion punktemitNachbarnVerbinden() werden die einzelnen Punkte als Nachbarn(Edges) hinzugefügt.
In der treppeOderAufzug() Funktion wird geprüft ob sich der Y-Wert (entspricht der Höhe) vom nächsten Pnkt unterscheidet, ist das der Fall liegt ein Aufzug oder eine Treppe vor. Enspricht die Verschiebung genau 1 oder -1 liegen die Punkte direkt übereinander und sid ein Aufzug,andernfalls ist es eine Treppe.

```
function importPathMesh(pfadZumOBJ){
    let objString = ObjInStringKonvertieren(pfadZumOBJ)
    let splitString = objSting.inStringArraySplitten(„\n“)
    let punkteUndEdgeData = punktDatenInPathMeshPoints(splitString)
    let pathMeshPoints = punkteMitNachbarnVerbinden(punkteUndEdgeData)
    treppeOderAufzugMarkieren(pathMeshPoints)
    return pathMeshPonts
}

punktDatenInPathMeshPoints(string){
    let points = []
    let edgeData = []
    let index = 0
    for (each in string){
        if (each[0] == „v“){
            points.push(neuer PathMeshPont(index-1, koordinaten))
            index++
        }
        if (each[0] == „l“){
            edgeData.push([ersteVertexID, zweiteVertexID])
        }
    return [points, edgeData]}
```

```
punkteMitNachbarnVerbinden([points, edgeData]){
    for (punkt in points){
        for(daten in edgeData){
            if (punkt.ID in daten gefunden){
                erstelle Edge(NachbarPunkt, entfernungBerechnen)
            }
        }
    }
    return pathMeshPoints
}

treppeOderAufzugMarkieren(punkte){
    wenn (richtungsVektor.Y zu einem Nachbarpunkt != 0){
        wenn (richtungsVektor.Y zu einem Nachbarpunkt == 1 oder -1){
            aufzug = true
        } else{
            treppe = true
        }
    }
}
```



Routenalgorithmus

Mustertext wird noch angepasst:
Beim Routenalgorithmus wurde der A*-Algorithmus ausgesucht mit dem man den kürzesten Weg inn zwischen zwei Knoten bestimmen kann. Dafür wird bei jedem Schritt der Knoten expandiert, dessen Kosten am geringsten sind. Um einen vollständigen weg am ende zu bekommen kann man die Knoten abspeichern und am ende invertieren.

Ausblick

- Fixen der Aktuellen Bus, Wegfindung im Dritten Stock, zurücksetzen der Kamera
- Optimieren des Algorithmus nach mMenschlichen präferenzen
- Hinzufügen von Infos zu den Räumen, welche Fächer finden statt, hat der Raum eine Beschreibung
- Kategorisieren der Räume: Arbeitsräume für Studenten, Labore, Organisatorisches
- Hinzufügen von anderen Gebäuden