Project 3

2019049716

A. Manipulate the camera in the same way as in Project1 using your Project1 code(10 pts).

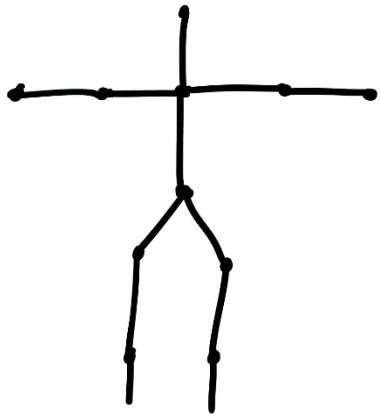
해당 부분은 원래의 코드를 그대로 가져왔습니다.

B. Load abvhfile and render it(110pts)

Drop callback을 사용해서 파일이 입력으로 들어오면 parsing을 했습니다.

들어온 순서대로 joint들의 이름을 저장하고, 해당 joint의 이름을 key로 해서 자신의 부모 노드의 이름을 알 수 있는 Parent, Channel, 그리고 offset들을 저장했습니다.

저는 모든 Link들을 한 점을 중심으로 그려 놓은 다음에 offset과 rotate 정보를 이용해서 joint에 그림을 그렸습니다.



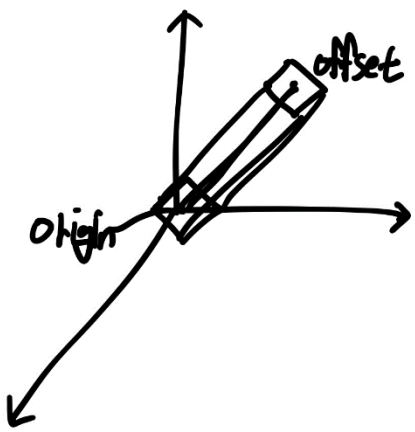
Rest pose 자세를 생각해보면, Root 노드인 Hip의 위치에서 3개의 Link가 나오는 것을 볼 수 있습니다. 즉, 부모 자식 사이의 offset 만큼의 Link를 부모 노드의 위치에서 그림을 그려주면 되는 것입니다.

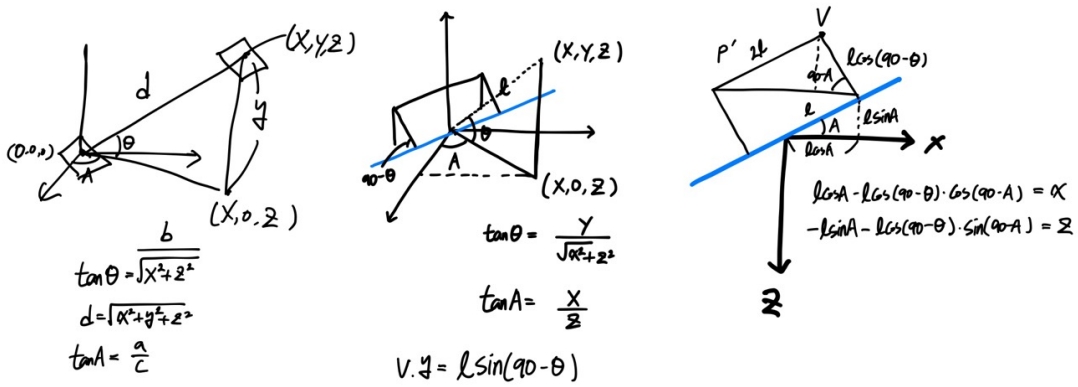
Root 노드의 offset을 원점으로 하여 원점에서부터 offset 만큼 떨어진 점을 잇는 선들 각각을 vao로 만들었습니다. 그러면 Link 수만큼의 vao가 만들어질 것입니다.

선으로 이루어진 캐릭터는 단순히 점과 점을 잇는 선을 그리는 것으로 만들 수 있었습니다.

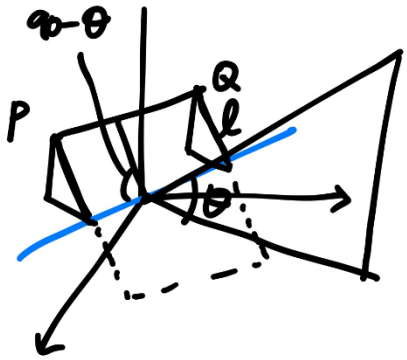
그런데 박스는 3차원 큐브를 그리는 것이므로 훨씬 복잡했습니다.

임의의 offset에 대해서 왼쪽의 그림처럼 큐브를 그려야 했기 때문입니다. 각 vertex의 좌표를 구하는 것이 까다로웠습니다.





위와 같은 계산을 통해 vertex의 좌표를 구할 수 있습니다.



왼쪽에서 P와 Q에대 한 좌표만 구하면 나머지 2개의 좌표는 원점에 대해 대칭이므로 쉽게 구할 수 있습니다.
그리고 반대편 면에 해당하는 4개의 vertex는 각 점에 대해서 offset을 vector로 해서 더해주면 구할 수 있습니다.

이렇게 box의 vertex들을 구할 수 있고, vertex 3개를 이용하면 외적을 통해 face의 normal도 구할 수 있습니다. 그렇게 얻은 vertex와 normal에 대한 정보를 이용해서 vao를 만들어주면 됩니다. 역시 Link의 수 만큼의 box_vao들이 만들어집니다.

생성한 vao를 각 조인트의 노드를 이용해서 매 frame마다 그려주면 됩니다

```
for element in allElements:
    if(element == Root): continue
    if(key_one == True):
        draw_node(vao_list_line[allElements.index(element) - 1], Nodes_one[parent[element]], P*V, MVP_loc, color_loc)
    elif(key_two == True):
        draw_node(vao_list_box[allElements.index(element) - 1], Nodes_two[parent[element]], P*V, MVP_loc, color_loc)
```

앞서 말했듯이 vao를 이용해서 렌더링할 때에는 부모 joint의 위치에서 그려주면 됩니다.

Key_one과 key_two는 key_callback에 set되어지는 변수로 1을 누르면 key_one이 활성화되어 선으로 이루어진 캐릭터가 렌더링되고, 2를 누르면 key_two가 활성화되어 박스로 이루어진 캐릭터가 렌더링됩니다.

Youtube: https://youtube.com/shorts/gG0Y7O5dsF8?feature=share