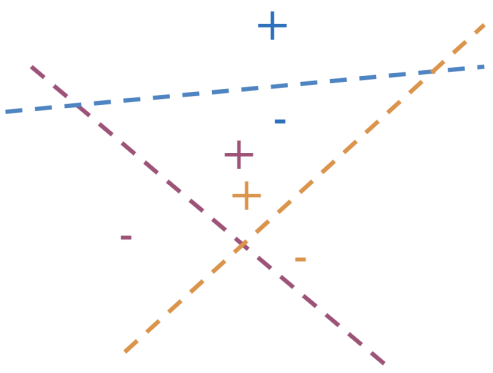
coursera

☰ **Item Navigation**

Multiple Planes

You can use multiple planes to get a single hash value. Let's take a look at the following example:



$$\mathbf{P}_1\mathbf{v}^T = 3, sign_1 = +1, h_1 = 1$$

$$\mathbf{P}_2\mathbf{v}^T = 5, sign_2 = +1, h_2 = 1$$

$$\mathbf{P}_3\mathbf{v}^T = -2, sign_3 = -1, h_3 = 0$$

$$hash = 2^0 \times h_1 + 2^1 \times h_2 + 2^2 \times h_3$$
$$= 1 \times 1 + 2 \times 1 + 4 \times 0$$
$$= 3$$

Given some point denoted by **v,** you can run it through several projections P_1, P_2, P_3 to get one hash value. If you compute $P_1 v^T$ you get a positive number, so you set $h_1 = 1$. $P_2 v^T$ gives you a positive number so you get $h_2 = 1$. $P_3 v^T$ is a negative number so you set h_3 to be 0. You can then compute the hash value as follows.

$hash = 2^0 \times h_1 + 2^1 \times h_2 + 2^2 \times h_3$

$= 1 \times 1 + 2 \times 1 + 4 \times 0 = 3$

Another way to think of it, is at each time you are asking the plane to which side will you find the point (i.e. 1 or 0) until you find your point bounded by the surrounding planes. The hash value is then defined as:

$hash_value = \sum_i^H 2^i \times h_i$

Here is how you can code it up:

```
def hash_multiple_plane(P_l,v):
    hash_value = 0
    for i, P in enumerate(P_l):
        sign = side_of_plane(P,v)
        hash_i = 1 if sign >=0 else 0
        hash_value += 2**i * hash_i
    return hash_value
```

P_l is the list of planes. You initialize the value to 0, and then you iterate over all the planes (P), and you keep track of the index. You get the sign by finding the sign of the dot product between v and your plane P. If it is positive you set it equal to 1, otherwise you set it equal to 0. You then add the score for the ith plane to the hash value by computing $2^i \times h_i$.

Mark as completed

Like **Dislike** **Report an issue**