Convex Optimization:

Inbar Lev Tov 316327246

The camera was positioned at origin – the far side of the screen, to allow maximum visibility.

I randomly sampled 5 points in a 3D coordination system – they would be the "original points". As you'll be able to see in the graph below, the points are rendered in red, and the vectors that connects them to the camera are rendered in blue.

I created the transformation matrix that consisted of rotation + translation (no scaling required). I wanted to rotate the points by 20 degrees and move them by 2 on the X axis and by 3 on the Y axis. When you look at the rotation matrix in particular, you realize that simply defining the degrees by which you want to rotate isn't enough – you have to calculate:

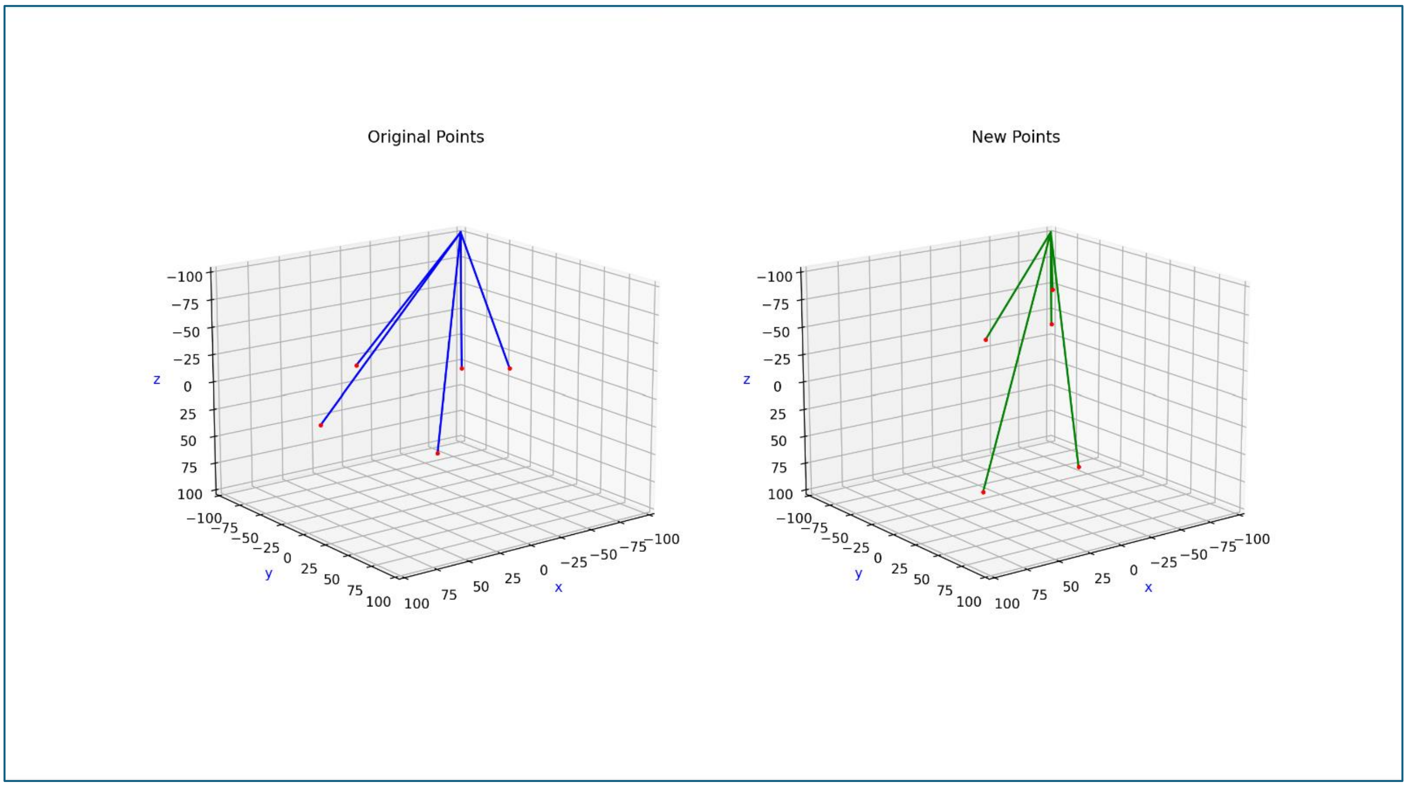
$$\cos(20^\circ) = 0.408$$

$$\sin(20^\circ) = 0.342$$

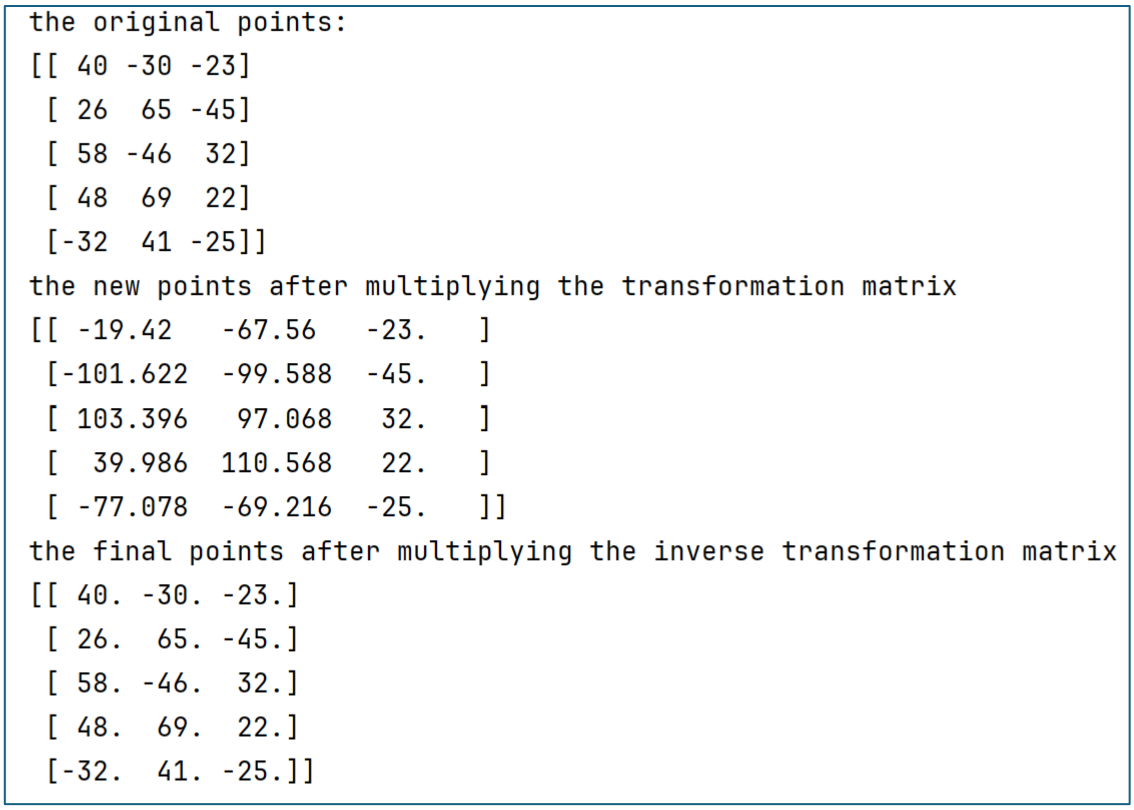
So now the full transformation composition is this:

$$\begin{pmatrix} \cos(x) & -\sin(x) & t_x \\ \sin(x) & \cos(x) & t_y \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.408 & -0.342 & 2 \\ 0.342 & 0.408 & 3 \\ 0 & 0 & 1 \end{pmatrix}$$

Now that we have the final transformation matrix, I multiplied it with each of the points and received new points in the same 3D coordination system. The new points are once again rendered in red, while the new vectors that connects them to the camera are rendered in green.



In order to return from the new points to the old ones, I calculated the inverse matrix and multiplied it with each of the new points. Here's an example for the entire process:



```
the original points:
[[ 40 -30 -23]
 [ 26  65 -45]
 [ 58 -46  32]
 [ 48  69  22]
 [-32  41 -25]]
the new points after multiplying the transformation matrix
[[ -19.42   -67.56   -23.    ]
 [-101.622  -99.588  -45.    ]
 [ 103.396   97.068   32.    ]
 [  39.986  110.568   22.    ]
 [ -77.078  -69.216  -25.    ]]
the final points after multiplying the inverse transformation matrix
[[ 40. -30. -23.]
 [ 26.  65. -45.]
 [ 58. -46.  32.]
 [ 48.  69.  22.]
 [-32.  41. -25.]]
```

Here the transformation matrix was fairly simple and the sampling size (only 5 points) was fairly small – so the cost we might have received by those changes is 0. With a larger sample size or alternatively an "uglier" transformation matrix, we would receive some cost and not manage to return to the same original points.

The next step would be: looking for a way to minimize the cost