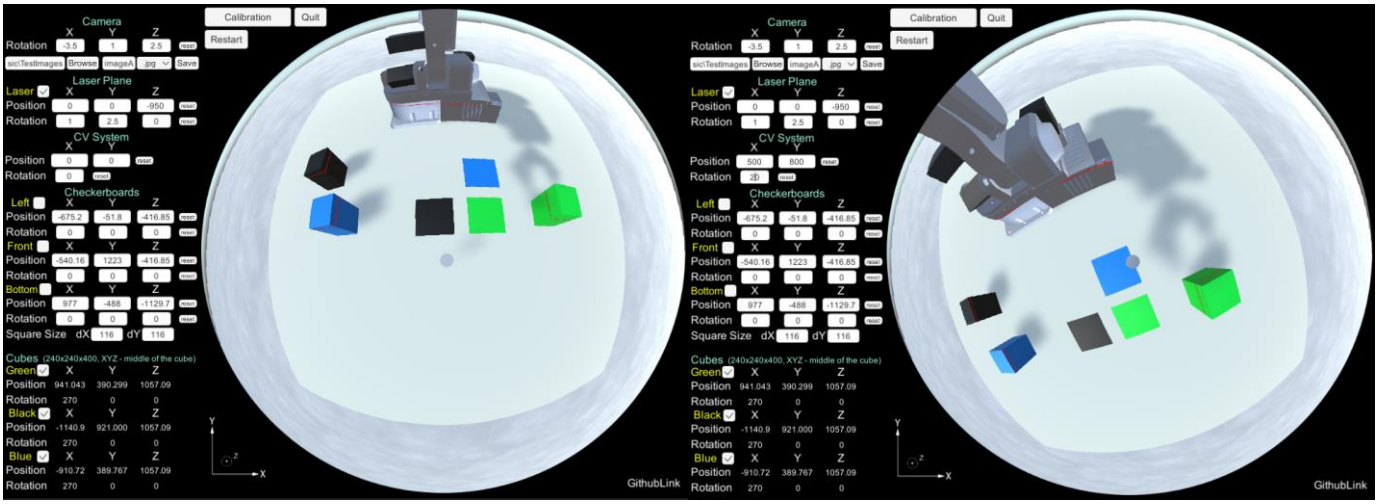
ГЛОБАЛЬНАЯ КАРТА

В этом руководстве мы узнаем, как построить глобальную карту со следующей конфигурацией системы:

- Камера повернута вокруг оси X на -3,5°, оси Y на 1° и вокруг оси Z на 2,5°;
- Лазерная плоскость повернута вокруг оси X на 1° и вокруг оси Y на 2,5°;
- Расстояние между камерой и плоскостью лазера 950 мм;
- После создания первого снимка мы переместим систему в другое положение, чтобы построить глобальную карту.



Процедура калибровки

Прежде всего, вам необходимо откалибровать камеру, если вы еще этого не сделали - сначала перейдите к руководству по калибровке.

Подготовка

Теперь нам нужно подготовить изображения с интересующим нас участком окружающей среды. Давайте повернем нашу камеру вокруг оси X на -3,5°, вокруг оси Y на 1° и вокруг оси Z на 2,5°; Лазерная плоскость вокруг оси X на 1° и вокруг оси Y на 2,5°. В результате наши изображения будут выглядеть так, как на картинке выше. Далее нам нужно переместить систему в другое место и сделать там снимок, см. рисунок выше.

!!! Запишите смещение и поворот системы, это понадобится для нашей программы Matlab. В случае выше: Смещение (500; 800) и Поворот (20).

В этом руководстве есть несколько дополнительных файлов Matlab, которые должны быть включены в папку проекта:

- C_calib_data
- cam2world
- compose_rotation

Лазерная сегментация

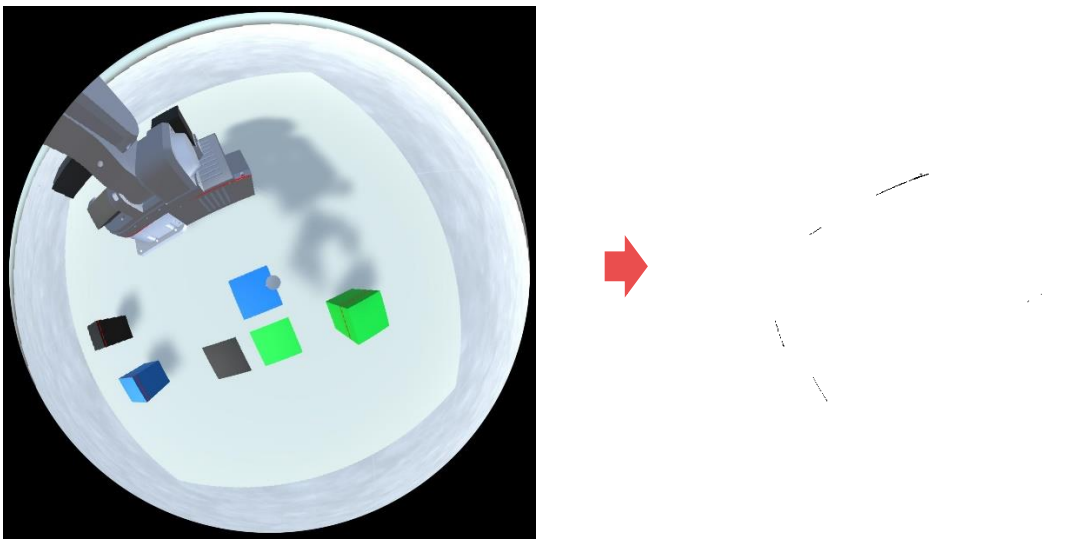
Наша первая функция Matlab будет связана с извлечением лазера. Мы будем использовать простую сегментацию изображения.
Загрузим наши изображения для дальнейшей работы

```
clc
clear all
image = imread('TestImages/image.jpg');
image1 = imread('TestImages/image1.jpg');
```

Теперь нам нужно создать новую функцию для извлечения лазера.

```
img = las_segm(image);
img1 = las_segm(image1);
```

После применения вышеуказанного кода у нас будет извлечена только часть лазера, которая показана на рисунках ниже:



```
function BW = las_segm(img)
% http://matlabtricks.com/post-35/a-simple-image-segmentation-example-in-
warning('off','Images:initSize:adjustingMag');
image = img;     % read image
[height, width, planes] = size(image);
rgb = reshape(image, height, width * planes);
imagesc(rgb);                    % visualize RGB planes
r = image(:, :, 1);              % red channel
g = image(:, :, 2);              % green channel
b = image(:, :, 3);              % blue channel
threshold = 100;                 % threshold value
imagesc(r < threshold);          % display the binarized image
blueness = double(r) - max(double(g), double(b));
imagesc(blueness);               % visualize RGB planes
mask = blueness < 78;
imagesc(mask);
% labels = bwlabel(mask);
R(~mask) = 255;
G(~mask) = 255;
B(~mask) = 255;
J = cat(3,R,G,B);
BW = ~mask;
% Skeletonization
% BW = im2bw(J,0.4);
% BW = bwmorph(BW,'skel',Inf);
imshow(~BW);
end
```

Обратите внимание, что значение «маски» blueness <78 было установлено для определенной интенсивности лазера, поэтому, если вы планируете изменить интенсивность лазера (другой источник), измените также данное значение.

Функция построения карты

Чтобы рассчитать расстояние от камеры до препятствий, нам нужно создать функцию.

```
load('Omni_Calib_Results.mat'); % Calib parameters
ocam_model = calib_data.ocam_model; % Calib parameters
```

Давайте настроим наши известные параметры, связанные с камерой, лазерной плоскостью и роботом:

```
camX =-3.5; % Camera parameters
camY = 1; % Camera parameters
camZ = 2.5; % Camera parameters
lasX = 1; % Laser Plane parameters
lasY = 2.5; % Laser Plane parameters
las_dist = 950; % Laser Plane parameters
CVsyst_x = 0; % CV System initial position
CVsyst_y = 0; % CV System initial position
CVsyst_rot = 0; % CV System initial rotation
CVsyst_x1 = 500; % CV System second position
CVsyst_y1 = 800; % CV System second position
CVsyst_rot1 = 20; % CV System second rotation
```

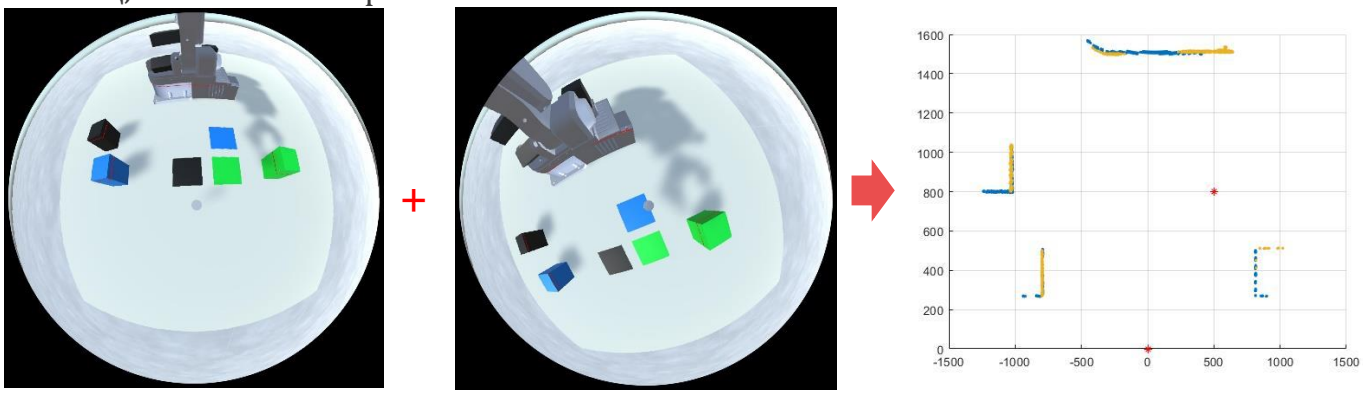
Теперь мы готовы начать процесс написания функции.

```
[x,y] = mapping(img,CVsyst_rot,CVsyst_x,CVsyst_y,camX,camY,camZ,lasX,lasY,...
    las_dist,ocam_model); % mapping function
[x1,y1] = mapping(img1,CVsyst_rot1,CVsyst_x1,CVsyst_y1,camX,camY,camZ,lasX,...
    lasY,las_dist,ocam_model); % mapping function
```

В результате получаем координаты пересечение лазера с препятствиями `[x,y]` и `[x1,y1]`. Далее построим карту:

```
figure;
scatter(x,y,5,'filled'); % Laser intersections, first image
hold on;
plot(CVsyst_x,CVsyst_y,'r*'); % CV System location, first image
scatter(x1,y1,5,'filled'); % Laser intersections, second image
plot(CVsyst_x1,CVsyst_y1,'r*'); % CV System location, second image
grid on;
```

Наконец, глобальная карта:

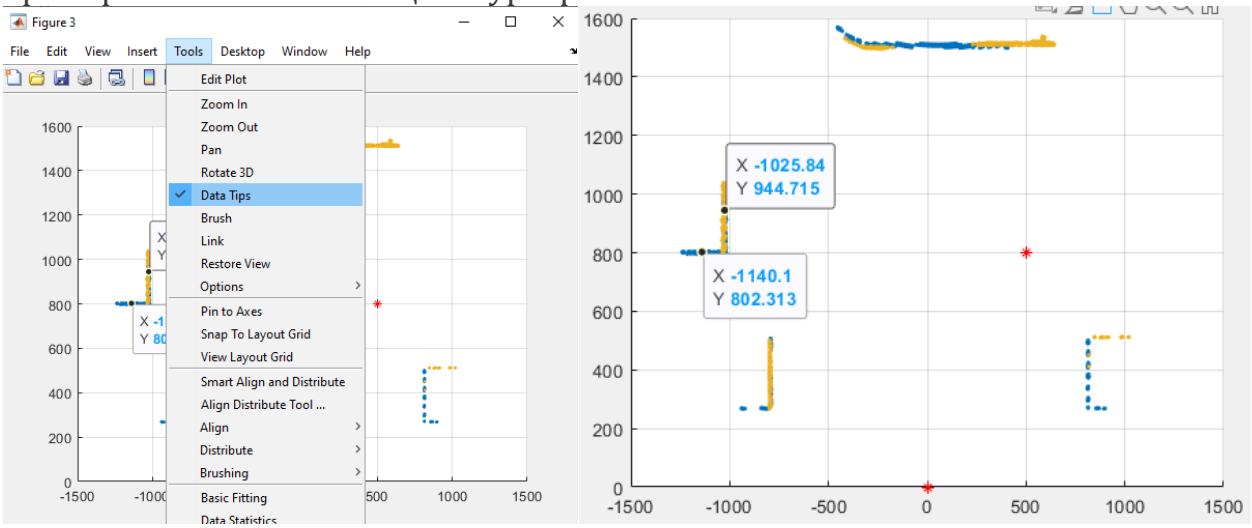


```
function [x,y] = mapping(image,cvsyst_rot,cvsyst_y,cvsyst_x,camY,camX,...
    camZ,lasY,lasX,las_dist,ocam_model)

[height,width] = size(image);
Z=las_dist;
a = 2;
x=[];
y=[];
t = [0;0;Z];
r = compose_rotation(-lasX, -lasY, 0);
r1 = compose_rotation(camX, camY, camZ);
r = r1*[r(:,1),r(:,2),t];
for i=1:height      % working image region
    for j=1:width
        if image(i,j)>0
            m=[i;j];       % image pixels
            M = cam2world(m,ocam_model); % transform from image plane to
the camera plane
            a1 = ?
            b1 = ?
            c1 = ?
            a2 = ?
            b2 = ?
            c2 = ?

            Y = ?
            X = ?

            M1=[X;Y;1];
            r2 = compose_rotation( ? );
            M1 = r2*M1; % CV System rotation

            y(1)=cvsyst_x; % CV System translation
            x(1)=cvsyst_y;

            y(a)=M1(1)+cvsyst_x;
            x(a)=-M1(2)+cvsyst_y;
            a=a+1;
        end
    end
end
end
```

Проверка результатов

После построения карты становится возможным сверить расстояния из эксперимента с реальными, вот пример того, как это можно сравнить.

Значение эксперимента мы можем рассчитать с помощью нашей глобальной карты. Проверьте расстояние с помощью курсора мыши.



	Черный куб - X, мм	Черный куб - Y, мм
Реальные значения	-1140,90	921,00
Экспериментальные значения	-1145,84	922,31
Погрешность	4,94	1,31