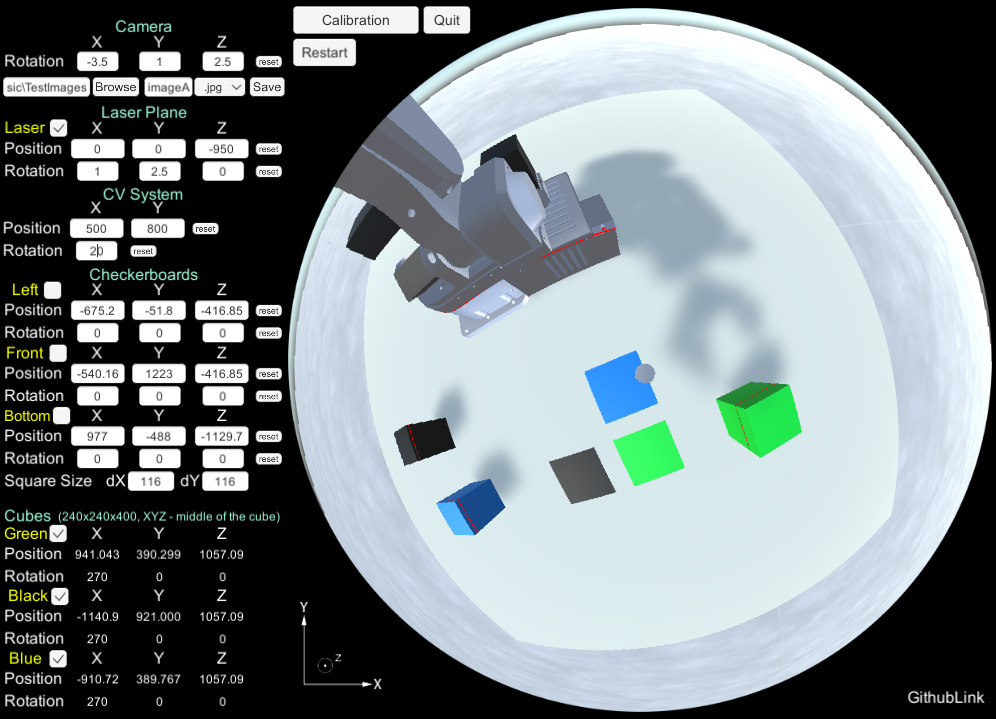
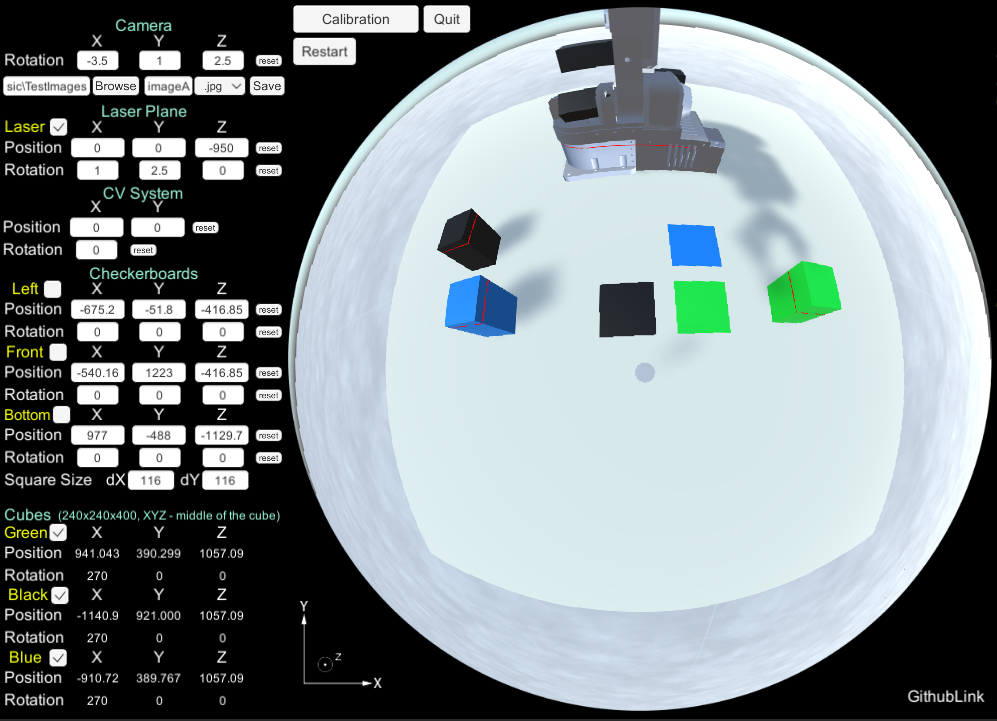
**Глобальная карта**

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

* Камера повернута вокруг оси X на -3,5o, оси Y на 1o и вокруг оси Z на 2,5o;
* Лазерная плоскость повернута вокруг оси X на 1o и вокруг оси Y на 2,5o;
* 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 -3,5o, вокруг оси Y на 1o и вокруг оси Z на 2,5o; Лазерная плоскость вокруг оси X на 1o и вокруг оси Y на 2,5o. 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

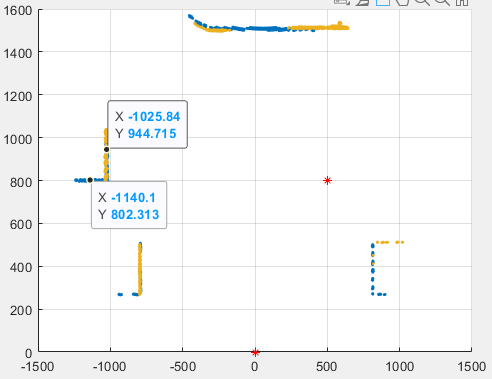
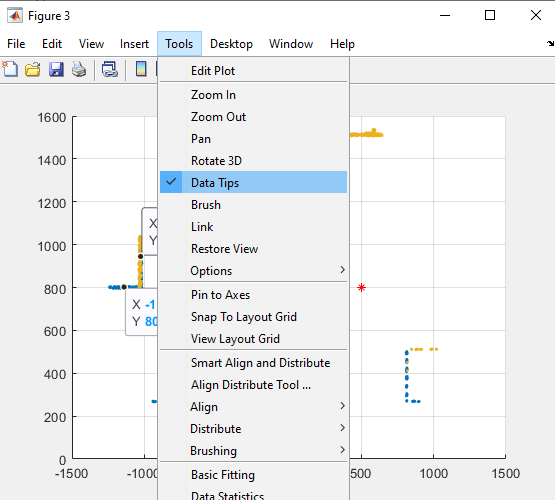
end

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 |