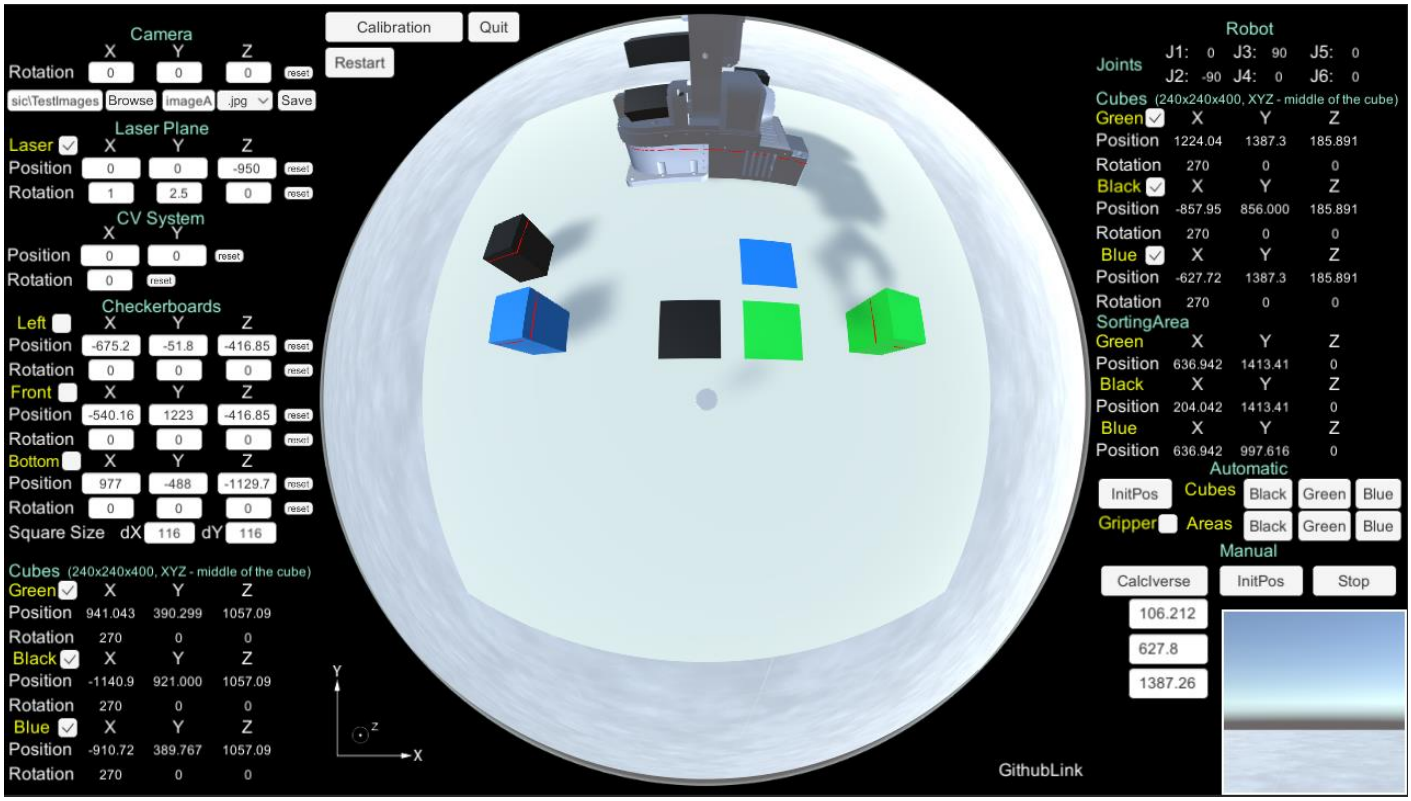
ОДИНОЧНАЯ КАРТА

В этом руководстве мы узнаем, как построить карту и провести проверку результатов со следующей конфигурацией системы технического зрения:

- Камера не поворачивается;
- Лазерная плоскость поворачивается вокруг оси X на 1,5° и вокруг оси Y на 2,5°;
- Расстояние между камерой и плоскостью лазера составляет 950 мм.



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 изображение с интересующей нас областью. Итак, давайте активируем все препятствия. Кроме того, давайте повернем нашу лазерную плоскость вокруг оси X на 1,5° и вокруг оси Y на 2,5°. В результате наше изображение будет выглядеть как на картинке выше.

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

- C_calib_data
- cam2world
- compose_rotation

Извлечение лазера

Наша первая функция Matlab будет связана с извлечением лазера. Мы будем использовать простую сегментацию изображения, для некоторых теоретических объяснений вы можете посетить веб-источник этого кода [1].

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

Теперь нам нужно создать новую функцию для извлечения лазера.

```
mg = las_segm(image);
```

```
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.

После применения вышеуказанного кода у нас будет извлечена только часть лазерной полосы, которая показана на рисунке ниже:

Построение карты

Чтобы рассчитать расстояние от камеры до препятствий, нам нужно создать функцию.
Прежде всего загрузим калибровочные параметры нашей камеры:

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

Во-вторых, давайте настроим наши известные начальные параметры, связанные с ориентацией лазерной плоскости:

```
x = 1.5; % Laser Plane parameters
y = 2.5; % Laser Plane parameters
las_dist = 950; % Laser Plane parameters
```

Мы не перемещали нашу систему, поэтому ее позиция равна нулю:

```
CVsyst_x = 0; % CV System position
CVsyst_y = 0; % CV System position
```

Теперь мы готовы начать процесс написания функции.

```
[x1,y1] = mapping(img,x,y,las_dist,ocam_model); % mapping function
```

```
function [x1,y1] = mapping(image,y,x,las_dist,ocam_model)
[height,width] = size(image);
Z=las_dist;
a = 1;
x1=[];
y1=[];
t = [0;0;Z];
r = compose_rotation(-x, -y, 0);
r = [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 = ?

            y1(a)= ?
            x1(a)= ?
            a=a+1;
        end
    end
end
end
```

В результате получаем координаты пересечения лазера с препятствиями `[x,y]`. Все, что нам нужно сделать сейчас, это построить карту:

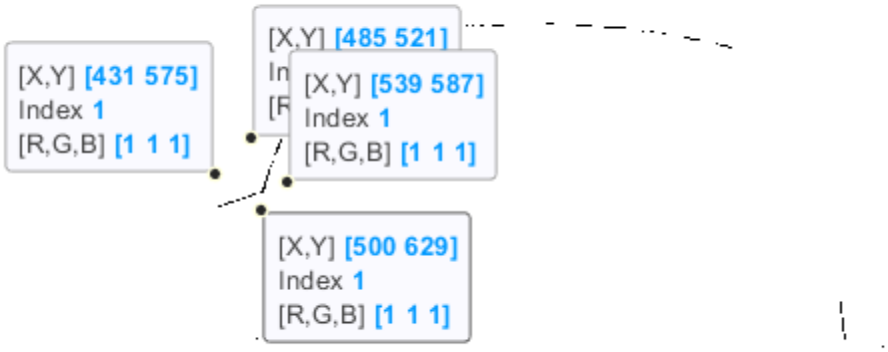
```
% Finally figure:
figure;
scatter(x1,y1,5,'filled'); % Laser intersections
hold on;
plot(CVsyst_x,-CVsyst_y,'r*'); % CV Systemlocation
grid on;
```

Теперь мы можем сравнить наше входное изображение с выходным. Как видите, выходное изображение содержит только информацию о расстоянии пересечения лазера с препятствиями:

Проверка результатов

После построения карты появляется возможность сравнивать расстояния из эксперимента с реальными. Для этого можно использовать построенную фигуру и курсор мыши. Однако лучше брать среднее значение среди всех лазерных точек, принадлежащих желаемому препятствию, потому что это поможет вам устранить некоторые помехи. Рассмотрим функцию для второго случая.

Прежде всего нам нужно найти интересующие нас координаты пикселей, чтобы найти лазерные точки, принадлежащие этой области, см. рисунок ниже:



После того, как координаты были определены, мы можем написать функцию, которая проецирует пиксельные координаты в координаты пространства (`Dist`).

```
function [Dist] = cube_dist(image,i_,j_,y,x,las_dist,ocam_model)

Z=las_dist;
a = 1;
x1=[];
y1=[];
t = [0;0;Z];
r = compose_rotation(-x, -y, 0);
r = [r(:,1),r(:,2),t];
for i=i_(1):i_(2)      % working image region
    for j=j_(1):j_(2)
        if image(j,i)>0
            m=[j;i];        % image pixels
            M = cam2world(m,ocam_model); % transform from image plane to
the camera plane
            a1 = ?
            b1 = ?
            c1 = ?
            a2 = ?
            b2 = ?
            c2 = ?

            Y = ?
            X = ?

            y1(a)= ?
            x1(a)= ?
            a=a+1;
        end
    end
end
Dist=[y1',x1'];
end
```

И, наконец, мы возьмем среднее значение каждой области

```
% Black Cube right
i=[485;539]; % working image region - column
j=[521;587]; % working image region - row
[C_left] = cube_dist(img,i,j,x,y,las_dist,ocam_model);
C_left = mean(C_left(:,1))-240/2;
% Black Cube bottom
i=[431;500]; % working image region - column
j=[575;629]; % working image region - row
[C_Up] = cube_dist(img,i,j,x,y,las_dist,ocam_model);
C_Up = mean(C_Up(:,2))+240/2;
```

Теперь мы можем сравнить значения:

	Черный куб - X, мм	Черный куб - Y, мм
Реальные значения	-1140,90	921,00
Экспериментальные значения	-1140,30	918,77
Погрешность	0,60	2,23

ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА:

1. Простой пример сегментации изображения в MATLAB. [В сети]. Доступно: http://matlabtricks.com/post-35/a-simple-image-segmentation-example-in- .