

1a)

Real time ball detection

Player and obstacle avoidance

Moving to ball location

Player and obstacle detection

Knowing court boundaries

Real time location computation

Detect Ball

Navigate

World awareness

1b)

Detect and track 3D location of ball using DPM + math

Detect and classify humans, and net, use DPM, SVM

cannyEdgedetector to find the boundaries of the court

Assume (0,0,0) is center court  
faceDirection(0,0,0)

Plot course to avoid net and humans

When ball Z=0 go fetch ball

FaceDirection(0,0,0) #assume 0,0,0 is center of court

Capture image

cannyEdgedetect(image)

save location of boundaries

detect net in image

(Nx, Ny, Nz) = calculateNetLocation(detection)

while(1):

capture image stream

detect ball in image

(X,Y,Z) = calculateBallLocation(detection)

if(Z <=0+eps)

from image stream detect humans

(Hx[], Hy[], Hz[]) = calculateHumanLocations(detection)

getBall(X,Y,Z, Hx, Hy, Hz); #i should pass the net location in here too

return to standing location

getBall(getLocation, avoidLocations)

if no avoid locations in the way:

moveToLocation(getLocation)

grabObjectAtLocation(getLocation)

victoryDanceat(getLocation)

else

avoid = detectCollision(avoidLocations)

moveToLocation(avoid [X] + 1m)

getBall(getLocation, avoidLocations-avoid)

2a)

data = getData([], 'test','list');

ids = data.ids(1:3);

for i= 1:3

calib = getData(ids{i}, 'test', 'calib');

image = getData(ids{i}, 'test', 'disp');

disparity = image.disparity;

figure;imagesc(disparity);

fT = calib.f\*calib.baseline;

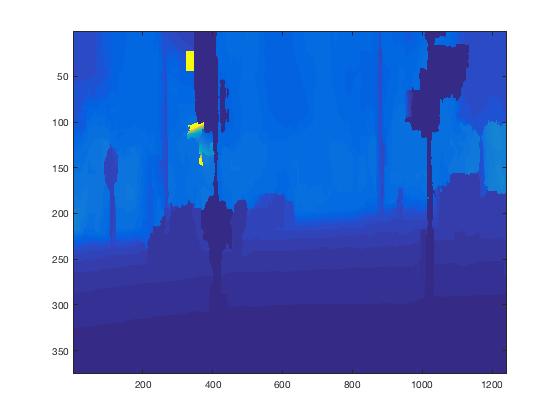
depth = fT./disparity;

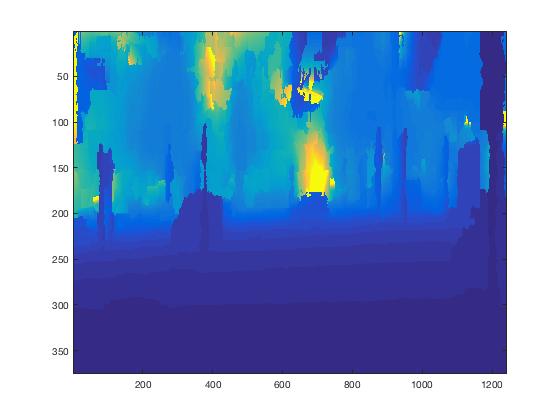
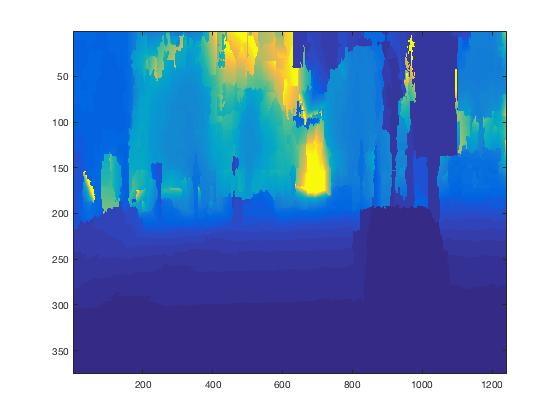
dfosho = depth;

dfosho(dfosho>255)=255+eps;

figure;imagesc(dfosho);

end





2b)

data = getData([], 'test','list');

ids = data.ids(1:3);

type = {'car', 'person', 'bicycle'};

thresh = {-0.5487,-0.7,-1.0916};

for j= 1:3

for i= 1:3

fdetector = sprintf('detector-%s',type{j});

data = getData([], [], fdetector);

model = data.model;

col = 'r';

cali

imdata = getData(ids{i}, 'test', 'left');

im = imdata.im;

f = 1.5;

imr = imresize(im,f); % if we resize, it works better for small objects

% detect objects

fprintf('running the detector, may take a few seconds...\n');

tic;

[ds, bs] = imgdetect(imr, model, thresh{j}); % you may need to reduce the threshold if you want more detections

e = toc;

fprintf('finished! (took: %0.4f seconds)\n', e);

nms\_thresh = 0.5;

top = nms(ds, nms\_thresh);

if model.type == model\_types.Grammar

bs = [ds(:,1:4) bs];

end

if ~isempty(ds)

% resize back

ds(:, 1:end-2) = ds(:, 1:end-2)/f;

bs(:, 1:end-2) = bs(:, 1:end-2)/f;

end;

figure;showboxesMy(im, reduceboxes(model, bs(top,:)), col);

fprintf('detections:\n');

ds = ds(top, :);

fname=sprintf('../data/test/results/%s-%s',ids{i}, type{j});

save(fname, 'ds');

end

end

Sample:

1.0e+03 \*

0.4997 0.1714 0.6435 0.2232 0.0010 0.0011

0.2270 0.1785 0.3719 0.2587 0.0030 0.0004

1.0677 0.1623 1.2449 0.2263 0.0010 0.0001

2c)

data = getData([], 'test','list');

ids = data.ids(1:3);

col = {'r', 'b', 'c'};

for i = 1:3

data = getData(ids{i}, 'test', 'ds');

imdata = getData(ids{i}, 'test', 'left');

im = imdata.im;

figure; axis ij; hold on

imagesc(im);

for c = 1:3

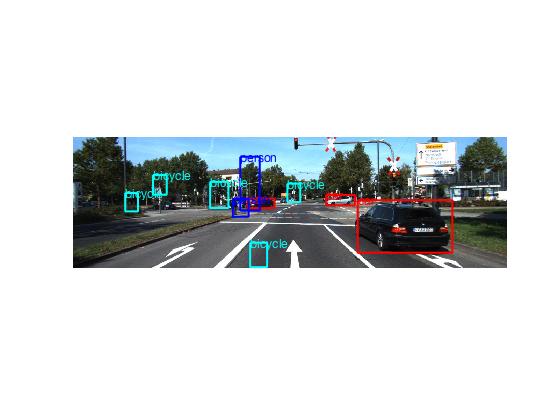
showboxesMy(im, data.dss{c}.ds(:,1:4), col{c});

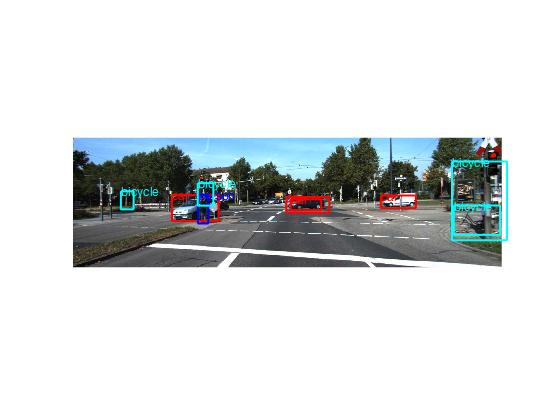
text(data.dss{c}.ds(:,1), data.dss{c}.ds(:,2), data.class{c},'Color',col{c},'FontSize',12);

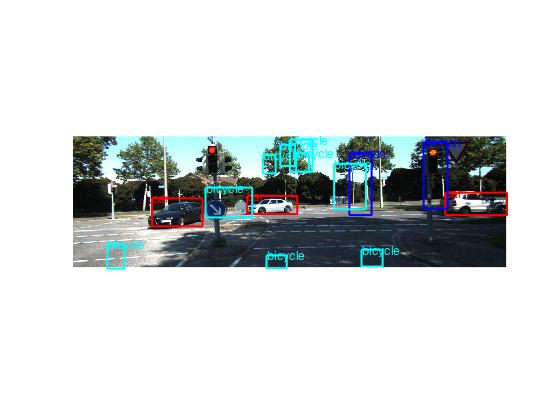
end

hold off;

end







2d)

data = getData([], 'test','list');

ids = data.ids(1:3);

col = {'r', 'b', 'c'};

for i = 1:3

data = getData(ids{i}, 'test', 'ds');

imdata = getData(ids{i}, 'test', 'left');

calib = getData(ids{i}, 'test', 'calib');

im = imdata.im;

depth = getDepth(ids{i});

for c = 1:3

for j = 1:size(data.dss{c}.ds,1)

x1 = round(data.dss{c}.ds(j,2):data.dss{c}.ds(j,4));

y1 = round(data.dss{c}.ds(j,1):data.dss{c}.ds(j,3));

x1 = x1(x1<=375);

y1 = y1(y1<=1242);

segmentdepth = depth(x1,y1);

Z = mode(round(segmentdepth(:)));

centers = [data.dss{c}.ds(j,1)+(data.dss{c}.ds(j,3)-data.dss{c}.ds(j,1))/2 data.dss{c}.ds(j,2)+(data.dss{c}.ds(j,4)-data.dss{c}.ds(j,2))/2];

X = (Z.\*(centers(1) - calib.K(1,3)))/calib.f;

Y = (Z.\*(centers(2) - calib.K(2,3)))/calib.f;

data.dss{c}.ds(j,7) = X;

data.dss{c}.ds(j,8) = Y;

data.dss{c}.ds(j,9) = Z;

end

ds = data.dss{c}.ds;

fname=sprintf('../data/test/results/%s-%s',ids{i}, data.class{c});

save(fname, 'ds');

end

end

2e)

data = getData([], 'test','list');

ids = data.ids(1:3);

col = {'r', 'b', 'c'};

for i = 1:3

data = getData(ids{i}, 'test', 'ds');

imdata = getData(ids{i}, 'test', 'left');

im = imdata.im;

depth = getDepth(ids{i});

plx = repmat((1:size(depth,1))', 1, size(depth,2));

ply = repmat((1:size(depth,2)), size(depth,1), 1);

Xmat = (depth.\*(plx - calib.K(1,3)))/calib.f;

Ymat = (depth.\*(ply - calib.K(2,3)))/calib.f;

segmask = uint8(zeros(size(depth)));

segim = uint8(zeros(size(im)));

for c = 1:3

X = data.dss{c}.ds(:,7);

Y = data.dss{c}.ds(:,8);

Z = data.dss{c}.ds(:,9);

data.dss{c}.ds(:,1:4)=floor(data.dss{c}.ds(:,1:4));

for j = 1:size(Z)

[rowz, colz] = find(depth>=Z(j)-3 & depth<=Z(j)+3);

[rowx, colx] = find(Xmat>=X(j)-3 & Xmat<=X(j)+3);

[rowy, coly] = find(Ymat>=Y(j)-3 & Ymat<=Y(j)+3);

indices = sub2ind(size(segmask), rowz, colz);

segmask(indices) = 1;

indices = sub2ind(size(segmask), rowx, colx);

segmask(indices) = 1;

indices = sub2ind(size(segmask), rowy, coly);

segmask(indices) = 1;

x1 = data.dss{c}.ds(j,2):data.dss{c}.ds(j,4);

y1 = data.dss{c}.ds(j,1):data.dss{c}.ds(j,3);

x1 = x1(x1<=375);

y1 = y1(y1<=1242);

for dim = 1:3

segim(x1 ,y1 ,dim)= im(x1 ,y1,dim).\*segmask(x1,y1);

end

end

end

figure; axis ij; hold on

imagesc(segim);

for c = 1:3

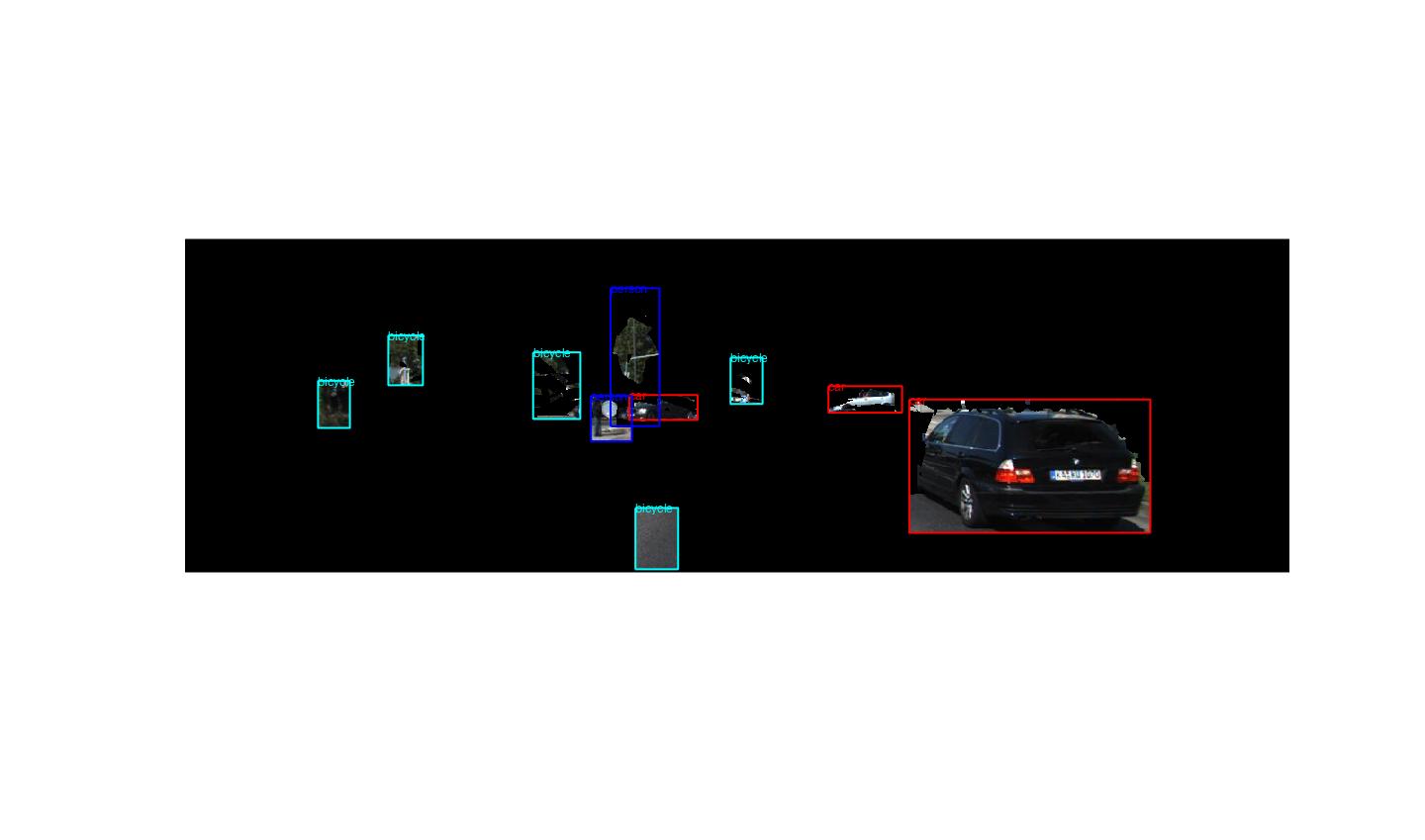
showboxesMy(im, data.dss{c}.ds(:,1:4), col{c});

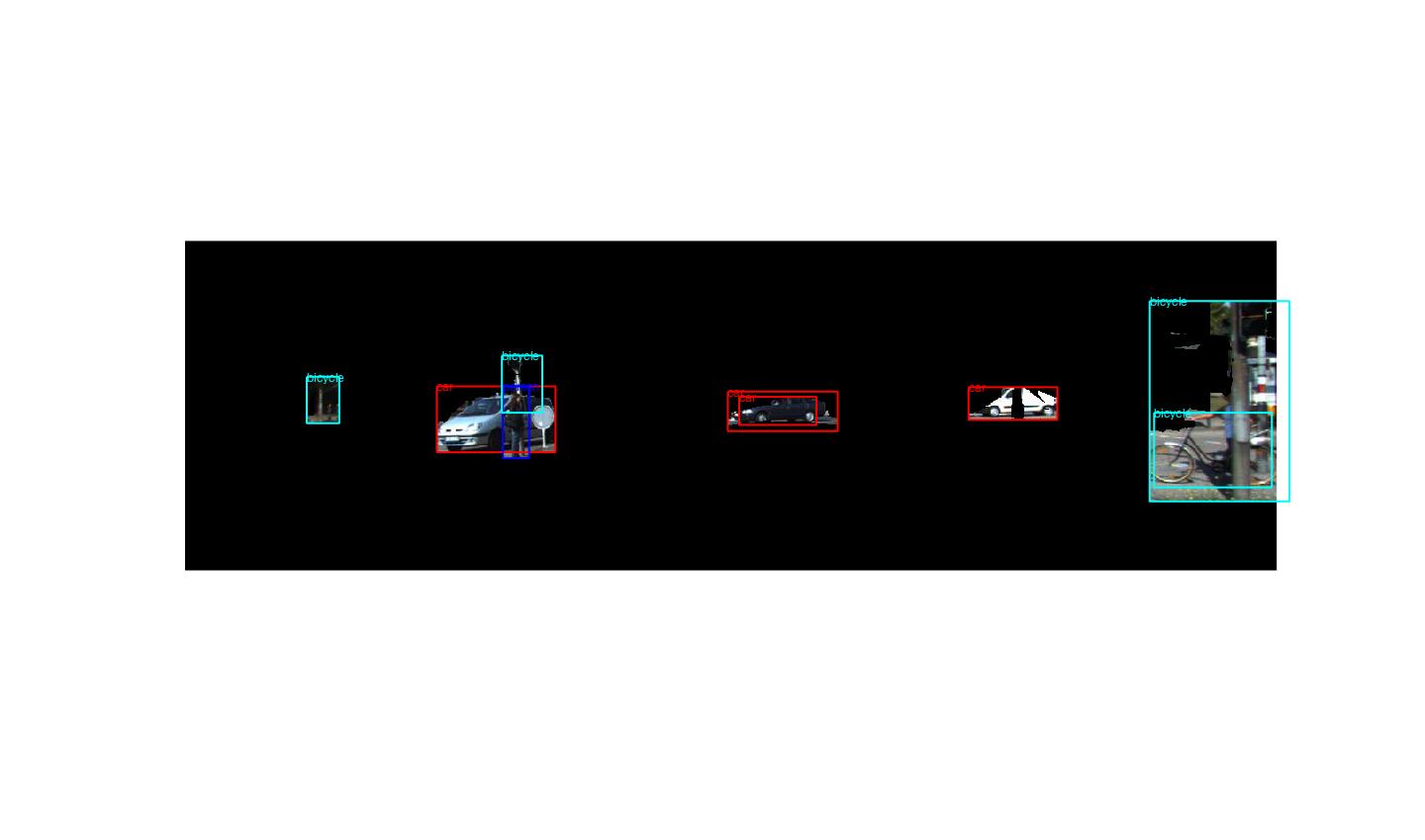
text(data.dss{c}.ds(:,1), data.dss{c}.ds(:,2), data.class{c},'Color',col{c},'FontSize',12);

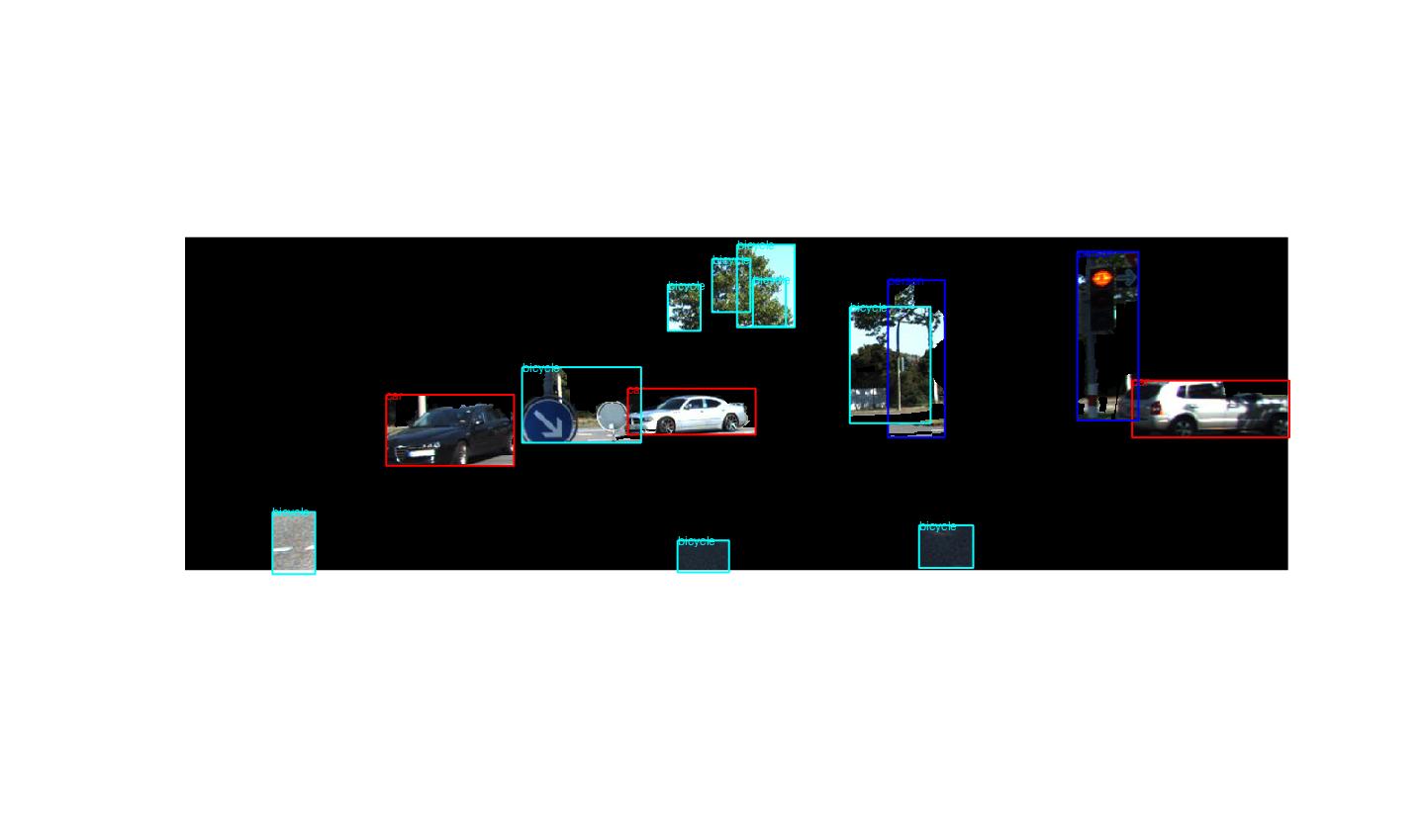
end

hold off;

end







2f)

data = getData([], 'test','list');

ids = data.ids(1:3);

col = {'r', 'b', 'c'};

for i = 1:3

data = getData(ids{i}, 'test', 'ds');

imdata = getData(ids{i}, 'test', 'left');

im = imdata.im;

d={};

for c = 1:3

X = data.dss{c}.ds(:,7);

Y = data.dss{c}.ds(:,8);

Z = data.dss{c}.ds(:,9);

for j = 1:size(Z)

d{end+1,1} = norm([X(j), Y(j), Z(j)]);

d{end,2} = data.class{c};

d{end,3} = X(j);

end

end

d = sortrows(d); %information about whatever is closest is presented first

for j = 1:size(d,1)

if d{j,3} >= 0, txt = 'to your right'; else txt = 'to your left'; end;

fprintf('There is a %s %0.1f meters %s \n', d{j,2}, d{j,3}, txt);

fprintf('It is %0.1f meters away from you \n', d{j,1});

end

end

## IMAGE1 from above page:

There is a bicycle -0.8 meters to your left

It is 7.2 meters away from you

There is a car 3.3 meters to your right

It is 7.8 meters away from you

There is a person -4.7 meters to your left

It is 26.4 meters away from you

There is a bicycle -15.9 meters to your left

It is 30.5 meters away from you

There is a car -3.4 meters to your left

It is 35.2 meters away from you

There is a car 10.2 meters to your right

It is 48.1 meters away from you

There is a bicycle 2.5 meters to your right

It is 78.1 meters away from you

There is a person -11.1 meters to your left

It is 78.9 meters away from you

There is a bicycle -41.0 meters to your left

It is 91.8 meters away from you

There is a bicycle -25.1 meters to your left

It is 98.3 meters away from you

## IMAGE2 from above page:

There is a bicycle 7.1 meters to your right

It is 11.5 meters away from you

There is a bicycle 8.6 meters to your right

It is 14.0 meters away from you

There is a bicycle -5.3 meters to your left

It is 17.8 meters away from you

There is a person -5.5 meters to your left

It is 17.9 meters away from you

There is a car -6.4 meters to your left

It is 19.1 meters away from you

There is a car 2.9 meters to your right

It is 32.1 meters away from you

There is a car 3.1 meters to your right

It is 32.2 meters away from you

There is a car 16.2 meters to your right

It is 38.6 meters away from you

There is a bicycle -64.5 meters to your left

It is 121.5 meters away from you

## IMAGE3 from above page:

There is a bicycle -0.2 meters to your left

It is 6.2 meters away from you

There is a bicycle 2.4 meters to your right

It is 7.6 meters away from you

There is a bicycle -1.8 meters to your left

It is 8.2 meters away from you

There is a bicycle -4.7 meters to your left

It is 8.6 meters away from you

There is a person 4.8 meters to your right

It is 9.3 meters away from you

There is a car -6.4 meters to your left

It is 16.4 meters away from you

There is a car 13.6 meters to your right

It is 22.6 meters away from you

There is a car -1.3 meters to your left

It is 25.0 meters away from you

There is a bicycle 0.4 meters to your right

It is 39.5 meters away from you

There is a bicycle 2.5 meters to your right

It is 40.6 meters away from you

There is a bicycle -2.6 meters to your left

It is 41.4 meters away from you

There is a bicycle 2.8 meters to your right

It is 41.5 meters away from you

There is a person 13.1 meters to your right

It is 46.0 meters away from you

There is a bicycle 12.3 meters to your right

It is 49.6 meters away from you