CV Project 2

多媒體工程所
宋秉一
9957513

Content

Figures

Tables

Rectification

Since we have the camera parameters from Project 1, we could rectify image using $K[R\ t]$ camera parameters. The image rectification is an idea that transforms epipolar lines to be parallel. The different images after transformation will on the same image plane. According to the epipolar geometry, the rectification image plane coordinate could be extracted from camera parameters.

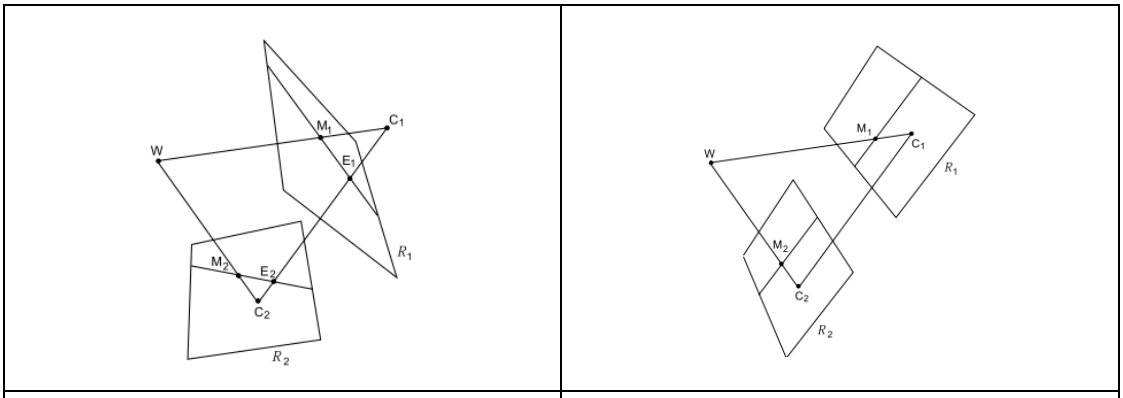

Image planes before rectificaion	Image planes after rectificaion

Figure 1.Rectificaion flow chart

Rectification Coordinate

In order to transform different cameras image into same image plane, we need a global coordinate which reference to camera 1 coordinate in our case.

- The new X axis is through the baseline cross two camera center $\frac{c_1-c_2}{\|c_1-c_2\|}$.
- The new Y axis is orthogonal to the new X and the Z axis of camera 1.
- The new Z axis is orthogonal to new X and new Y.

Let $K_1[R_1\ t_1]$ and $K_2[R_2\ t_2]$ be the camera 1 and camera 2 parameters. The camera center $t_1,\ t_2$ is now in the camera 1 and the camera 2 coordinate. In order to get the baseline vector, we need to transform them into the same world coordinate $c_1,\ c_2$.

$$\begin{cases} R_1 c_1 + t_1 = 0 \\ R_2 c_2 + t_2 = 0 \end{cases}$$

$$\begin{cases} c_1 = -{R_1}^{-1} t_1 \\ c_2 = -{R_2}^{-1} t_2 \end{cases}$$

Because the Rotation matrix is orthogonal, the inverse could be easily presented as its transpose. Now we have the camera centers in the world coordinate.

$$\begin{cases} c_1 = -{R_1}^{T} t_1 \\ c_2 = -{R_2}^{T} t_2 \end{cases}$$

Now we have the baseline vector $c_1 - c_2$. Divide the baseline length to get the new X unite vector axis. Since the Z axis of camera 1 could be got from $R_1(3,:)$, we have the new Y and the new Z axis shown as following.

$$X = \frac{c_1 - c_2}{\|c_1 - c_2\|}$$
$$Y = \frac{X \times R_1(3,:)}{\|X \times R_1(3,:)\|}$$
$$Z = \frac{X \times Y}{\|X \times Y\|}$$

Rectifying Transformation

We treat the rectification as a projection mapping since we have the rectification coordinate from pervious section. We set the rectification camera parameters as following.

$$K = \frac{K_1 + K_2}{2}$$
$$R = [X\ Y\ Z]$$

The rectification camera intrinsic parameter is set to be no skew (skew term = 0) since we want the epipolar lines parallel with image frame. This is important for the next part to get the disparity scanning line.

$$K(1,2) = 0$$

The projection matrix T projects 2D points from camera image to rectification image. Let p present the 2D points in camera image, and q present the 2D points in rectification image, λ be the arbitrary projection scalar.

$$\lambda q = Tp$$
$$\begin{cases} T_1 = KR(K_1 R_1)^{-1} \\ T_2 = KR(K_2 R_2)^{-1} \end{cases}$$

Warping Boundary

Now we have the rectifying matrix for warping. Before applying warping, we need to check the boundary of two image projection on the rectification image.

Let the size of image 1 and image 2 be $\mathrm{W}_1,\ \mathrm{H}_1,\ \mathrm{W}_2,\ \mathrm{H}_2$. Forward Project the image boundary points to the rectification image.

$$\lambda P_1 = \mathrm{T}_1 \begin{bmatrix} 0 & \mathrm{W}_1 & 0 & \mathrm{W}_1 \\ 0 & 0 & \mathrm{H}_1 & \mathrm{H}_1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\lambda P_2 = \mathrm{T}_2 \begin{bmatrix} 0 & \mathrm{W}_2 & 0 & \mathrm{W}_2 \\ 0 & 0 & \mathrm{H}_2 & \mathrm{H}_2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\begin{cases} x_M = \max[P_{1x}\ P_{2x}] \\ x_m = \min[P_{1x}\ P_{2x}] \\ y_M = \max[P_{1y}\ P_{2y}] \\ y_m = \min[P_{1y}\ P_{2y}] \end{cases}$$

The projection bounding box from $(x_m,\ y_m)$ to $(x_M,\ y_M)$.

Backward Warping

Instead of using the forward warping, we use the backward warping to avoid the hole with no projection position due to the integer resolution of image.

Let the rectified image be $\mathrm{E}_1, \mathrm{E}_2$ and the original image be $\mathrm{I}_1, \mathrm{I}_2$. This idea can be shown with a simple pseudo code.

For $(x_m,\ y_m)$ to $(x_M,\ y_M)$

  $\lambda P_1 = \mathrm{T}_1^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

  $\lambda P_2 = \mathrm{T}_2^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

  if P_1 is in bound of image 1

    $\mathrm{E}_1(x, y) = \mathrm{I}_1(P_1)$

  end

  if P_2 is in bound of image 2

    $\mathrm{E}_2(x, y) = \mathrm{I}_2(P_2)$

  end

end

Figure 2.The rectified images (left image5, right image7).

We can use the epipolar lines to validate the rectification result. Each epipolar line should through the same point in two images.

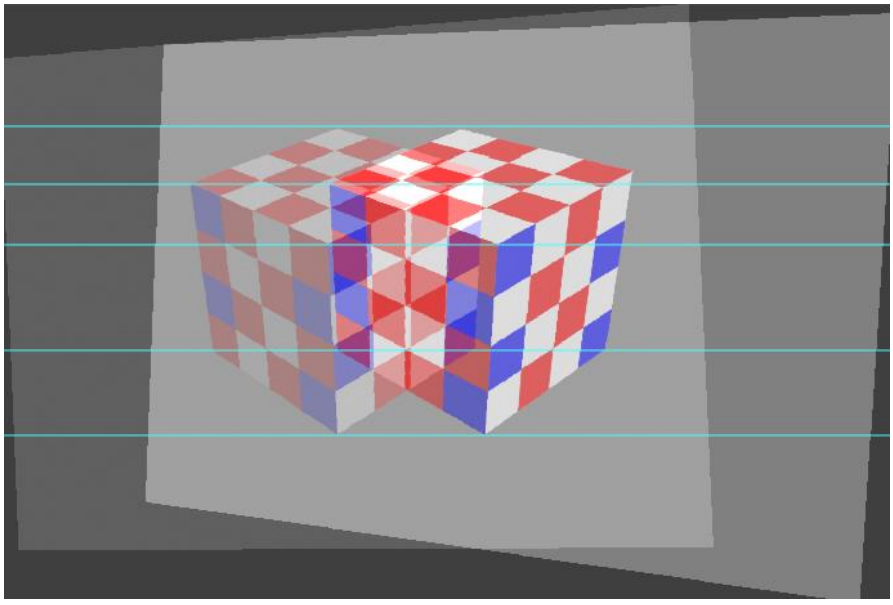

Figure 3.The rectified image with epipolar lines

Disparity map

After rectifying images, we can extract the depth from rectified images. This work can be partition into two parts. First part is matching position using dynamic programming, and the second part is computing the disparity map.
By applying a horizontal scanning line in each image with same y axis value, we can match the segment for each scanning line in image.

Segmentation

We use K-means for image labeling. Let K=4 for three colors on the target and one color be the background. Using K-means clustering method, we set the initial center

vectors manually and applying two round clustering. Select the best round result to be the labeled data.

Figure 4.The labeled images using K-means clustering.

Matching Cost

With horizontal scanning line on labeled image, we compare each segment with other image segment using standard deviation. There are three cost and the cost function types which includes match-two, match-left, match-right.

Let $a_1, a_2 \dots a_k$ and $b_1, b_2 \dots b_l$ be the intensity of left segment i and right segment j. C_P is cost for match-two, C_H is cost for match-right, C_V is cost for match-left.

$$\mathrm{m} = \frac{1}{2}(E(a) + E(b))$$

$$\sigma_{i,j}^2 = \frac{1}{2}\left(\frac{1}{k}\sum_{n=1}^{k}(a_n - m)^2 + \frac{1}{l}\sum_{n=1}^{l}(b_n - m)^2\right)$$

$$\begin{cases} \mathrm{C}_P = \sigma_{i,j}^2\sqrt{k^2 + l^2} \\ \mathrm{C}_V = k * f\left(\frac{1}{2}\left(\sigma_{i,j-1}^2 + \sigma_{i,j+1}^2\right), th\right) where\ th = \max\{\sigma_{i,j-1}^2, \sigma_{i,j+1}^2\} \\ \mathrm{C}_H = l * f\left(\frac{1}{2}\left(\sigma_{i-1,j}^2 + \sigma_{i+1,j}^2\right), th\right) where\ th = \max\{\sigma_{i-1,j}^2, \sigma_{i+1,j}^2\} \end{cases}$$

$$f(\sigma^2, th) = 2 * th - \sigma^2$$

Dynamic Programming

The dynamic programming method provides a minimum cost matching relation of segments on the scanning line of two images. We generate two table for recording the matching-route and matching-cost. Define the route code: match-two = 1, match-left = 2 and match-right = 3.

Let $D(i,j)$ presents the best matching-cost match segment i to segment j which we already consider the left segments $1, 2 \ldots i$, and the right segments $1, 2 \ldots j$. The recursively function as below. The $R(i,j)$ present where the minimum cost route is.

$$D(i,j) = \min \begin{cases} D(i-1,j) + C_V(i,j) \\ D(i-1,j-i) + C_P(i,j) \\ D(i,j-1) + C_H(i,j) \end{cases}$$

The desired cost is $D(k,l)$. In $D(k,l)$, we have already considered the route cost from left 1 to k segments matching with right 1 to l segments. Record the minimum cost direction in the matching-route table using the defined route code. We can back trace the matching route from $R(k,l)$.

1	3	3	3	3	2	2	2	1	2
2	1	1	1	3	1	3	1	1	2
2	1	1	3	1	3	1	1	3	2
2	1	2	1	3	1	3	1	3	2
2	2	1	2	1	3	1	1	3	2
2	1	2	1	2	1	3	1	3	2
2	2	1	2	1	2	1	1	3	2
2	1	1	1	1	1	1	1	3	2
3	3	3	3	3	3	3	3	3	1

Table 1.A sample of matching-route table

Disparity Computing

Once we have the minimum cost matching relation between two line segments, we cloud get the disparity from the corresponding matching points. The disparity is compute only when the matching-route is match-two case ($R(i,j) = 1$).

Due to the integer resolution of image, the disparity is interpolated by applying linear interpolation between segment start point and end point. The disparity map $Disp(m,n)$ is reference to left image in our project which m is the height of image and n is the width of image.

The disparity computing pseudo code

for scanning line $\text{row} = 1 \sim m$

If $R(i,j) = 1$

1. Get start point S_i and end point E_i of left segment i
2. Get start point S_j and end point E_j of right segment j
3. Compute the disparity $\text{S}_j - \text{S}_i$ and $\text{E}_j - \text{E}_i$

4. $Disp(row, \mathrm{S}_i) = \mathrm{S}_j - \mathrm{S}_i$
5. $Disp(row, \mathrm{E}_i) = \mathrm{E}_j - \mathrm{E}_i$
6. Applying linear Interpolation from $Disp(row, \mathrm{S}_i)$ to $Disp(row, \mathrm{E}_i)$

end

end

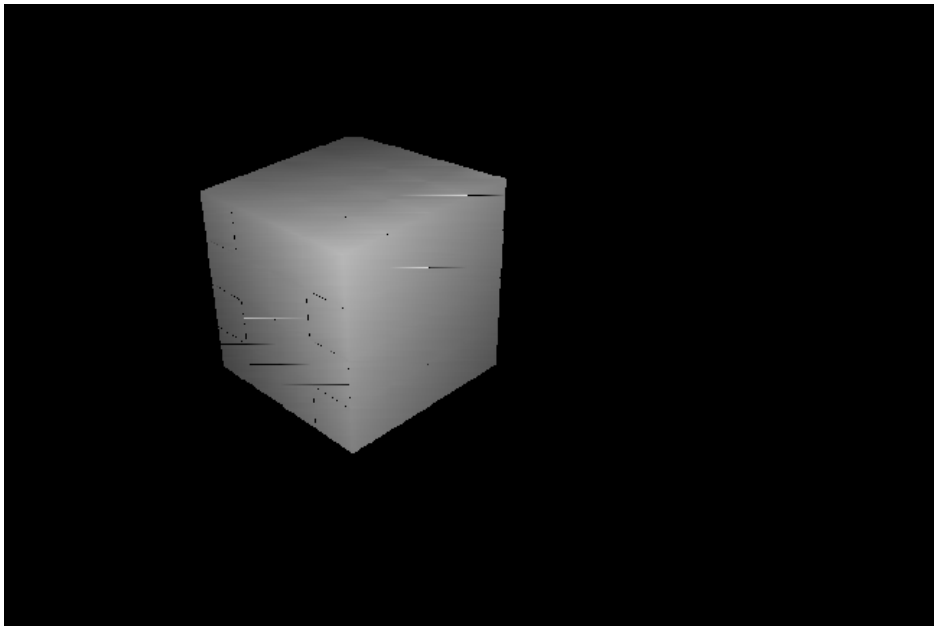

Figure 5.The Disparity map of left image without any refinement

Disparity Refinement

The dynamic programming method will give a minimum cost route no matter the route is right or wrong. The result shows that the disparity is not perfectly continuous. In order to get a reasonable disparity, we need to do some refinement on the disparity map.

Put a vertical scanning line on image and check the gradient of each pixel. The pixel is considered broken when the gradient is not continuous and we applying linear interpolation with it.

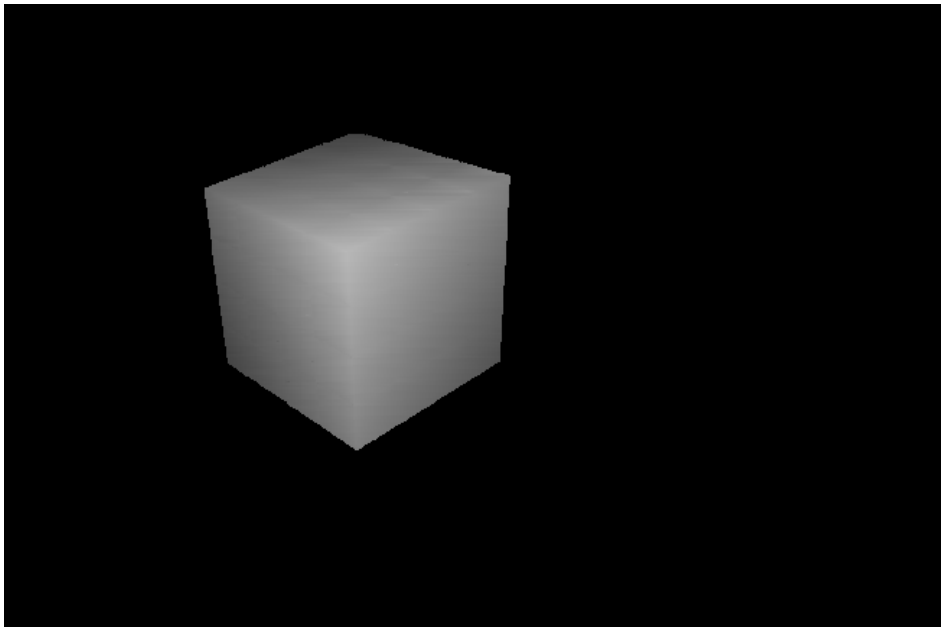

Figure 6.The Disparity map of left image with continuous refinement

Anaglyph image

Generate the right disparity map

The disparity map is referenced from left image in our project. We want to get the right view image referenced from the right disparity map. The right disparity map is the left position add left disparity. Before starting to compute the right disparity map, we have to adjust the left disparity to fit the human eyes. With the knowledge of the eye distance is 7cm and the original disparity is scaled by the baseline length; we could scale the disparity by a defined scalar.

$$S = \frac{7}{\|c_1 - c_2\|}$$

Now we apply the scalar with left disparity map.

$$DispLeft = DispLeft * S$$
$$DispRight = Left\ image\ Position + DispLeft$$

$$D_L \leftarrow D_L * S$$
$$\sout{D_R} \leftarrow P_L + D_L$$

****

$$P_R \leftarrow P_L + D_L$$

**** $I_{R\text{-}new\ base}(P_R) = I_L(P_L)$

The right disparity is obtained but with some holes in it due to the forward transform; we could interpolate holes with the same method of pervious section.

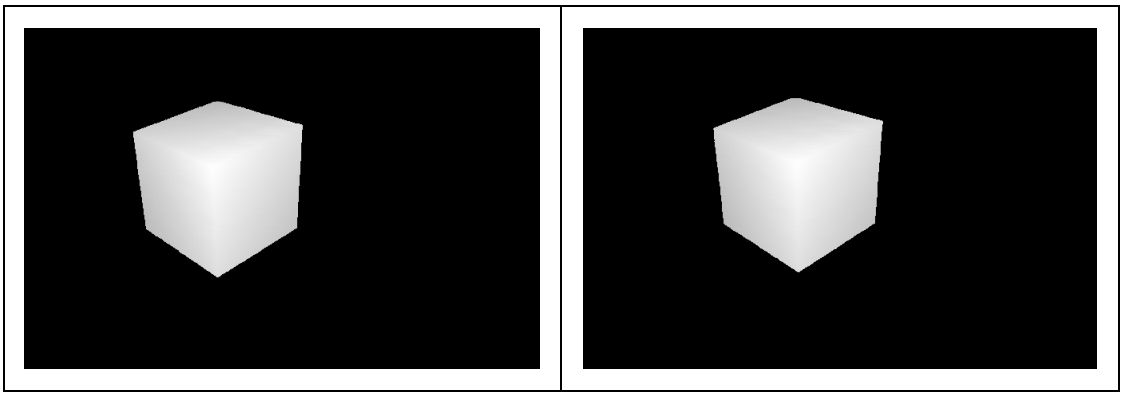

Figure 7.The scaled left disparity map and right disparity map.

Backward coloring

We use backward warping method to obtain the right view color. The correspondent position on left image is computed using the follow method.

$$\text{Right Image } = \text{Left Image}(Right\ image\ Position - DispRight)$$

$$I_R(P_R) = I_L(P_R - D_R)$$

$\text{Right Image } (Right\ image\ Position) =$

$\text{Left Image}(Right\ image\ Position - (Left\ image\ Position + DispLeft))$

$= I_L(P_R - P_L - D_L) = I_R(P_R)$

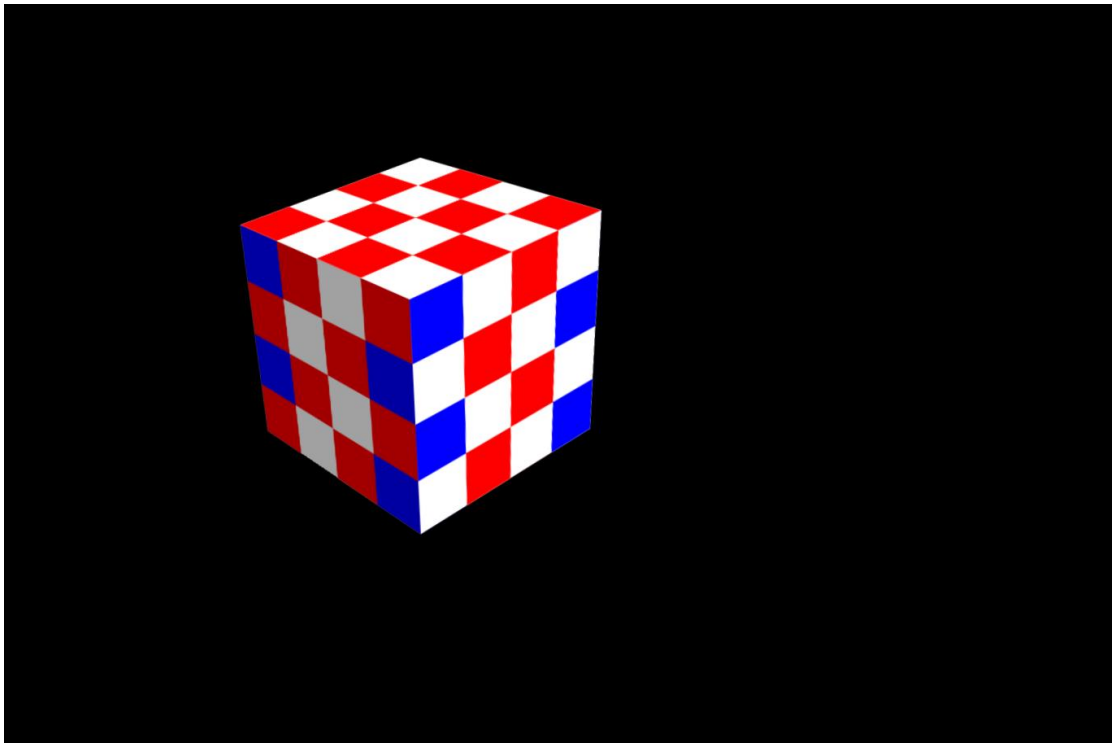

Figure 8.The left view image.

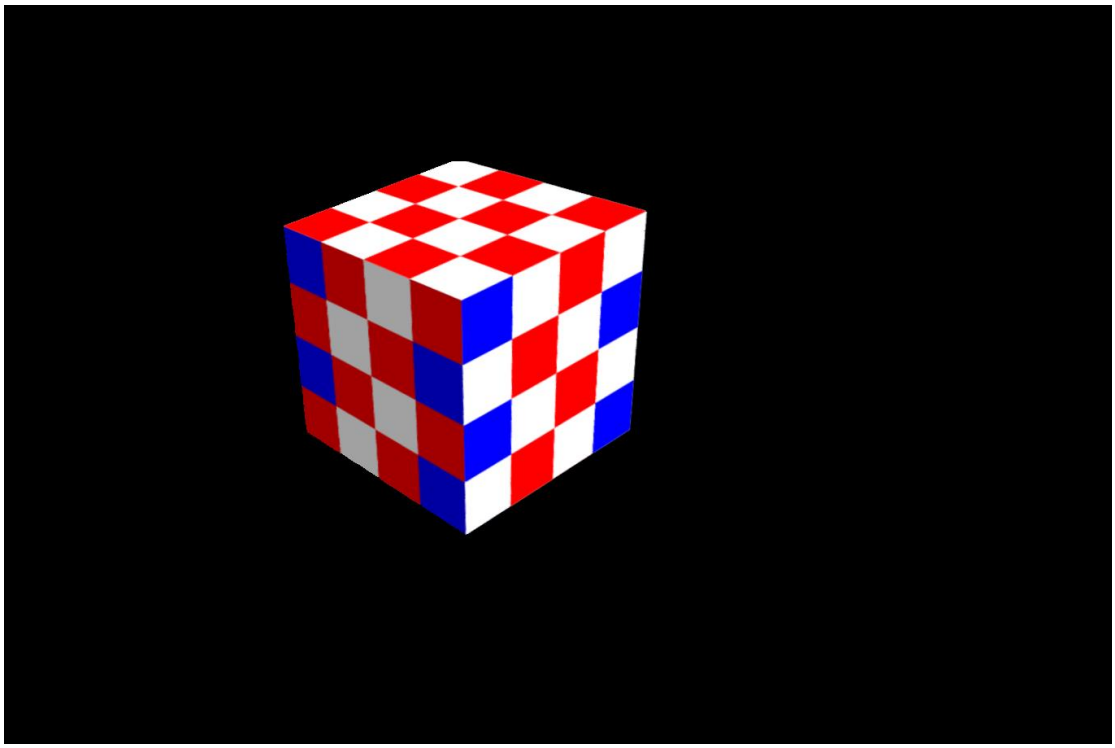

Figure 9.The right view image.

Blending two anaglyph images

We want to see the anaglyph image using the red-green or red-cyan glasses; blending the two view anaglyph images using the follow rule.

$$\text{anaglyph image} = \text{left view with R channel} + \text{right view with GB channel}$$

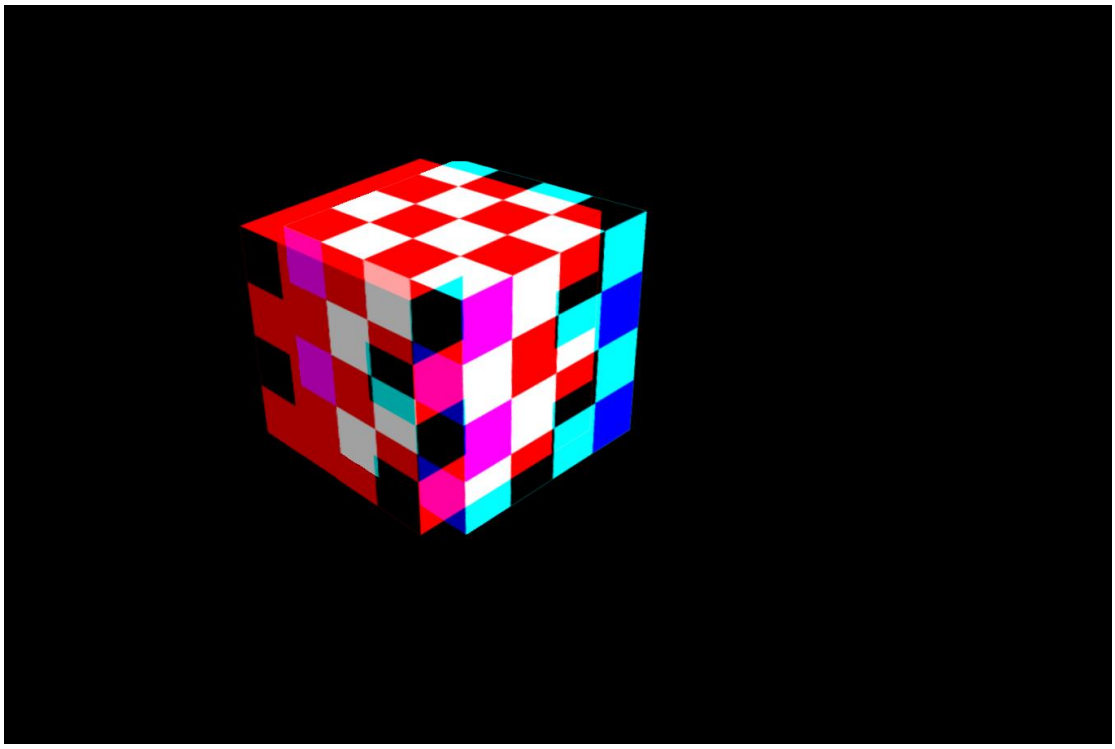

Figure 10.The anaglyph image.

Reference

[1] A. Fusiello, E. Trucco and A. Verri, “A compact algorithm for rectification of stereo pairs.” *Machine Vision* Appl. 12 1 (2000), pp. 16–22.

[2] Y. Ohta and T. Kanade, “Stereo by intra- and inter-scanline search using dynamic programming,” *IEEE Trans. Pattern Anal. Machine Intell*., vol. PAMI-7, (1985) pp. 139-154.