A Fast Marching Formulation of Perspective Shape from Shading under Frontal Illumination: Source Code

~ User Guide ~

The zip file consists of six C program files for users to construct their customized perspective SfS application based on the algorithm published at Pattern Recognition Letters "A Fast Marching Formulation of Perspective Shape from Shading under Frontal Illumination". In addition, four sets of depth, image and project files including: *four mountains*, *sphere*, *vase* and *mozart* are provided as test examples.

A. Description of major data files

B. Format of Depth Map file (e.g. 4mountains_psp_depth.txt)

[Number of row N_r] [Number of column N_c]
[Depth $D_{1,1}$ at (u_1, v_1)]
[Depth $D_{1,2}$ at (u_1, v_2)]
…
[Depth $D_{i,j}^{*}$ at (u_i, v_j)]
…
[Depth $D_{Nr,Nc}$ at (u_{Nr}, v_{Nc})]**

** Undefined depth is represented as 999999.0,*
*** Depths are arranged in Raster scanning order*

2. Format of Intensity Map file (e.g. 4mountains_psp_00_image.txt)

[Number of row N_r] [Number of column N_c]
[Intensity $I_{1,1}$ at (u_1, v_1)]
[Intensity $I_{1,2}$ at (u_1, v_2)]
…
[Intensity $I_{i,j}^{\#}$ at (u_i, v_j)]
…
[Intensity $I_{Nr,Nc}$ at (u_{Nr}, v_{Nc})]$^{\#\#}$

[#] *Undefined depth is represented as 999999.0,*

[##] *Depths are arranged in Raster scanning order*

3. Format of Shape from Shading Project file **(e.g. 4mountains_psp_Frontal_FMM.txt)**

[Filename of the perspective projected depth map]

[Undefined depth representative D_u][1]

[Depth Bias D_B][2]

[Number of Singular Point N_p]

[p_1 q_1] (Singular Point $S_1 = (p_1, q_1)$)

[p_2 q_2] (Singular Point $S_2 = (p_2, q_2)$)

…

[p_{Np} q_{Np}] (Singular Point $S_{Np} = (p_{Np}, q_{Np})$)

[Focal Length]

[Rendering Mode M_r][3]

[Number of rendering parameter N_r]

[Oblique angle α][4]

[Oblique angle β][4]

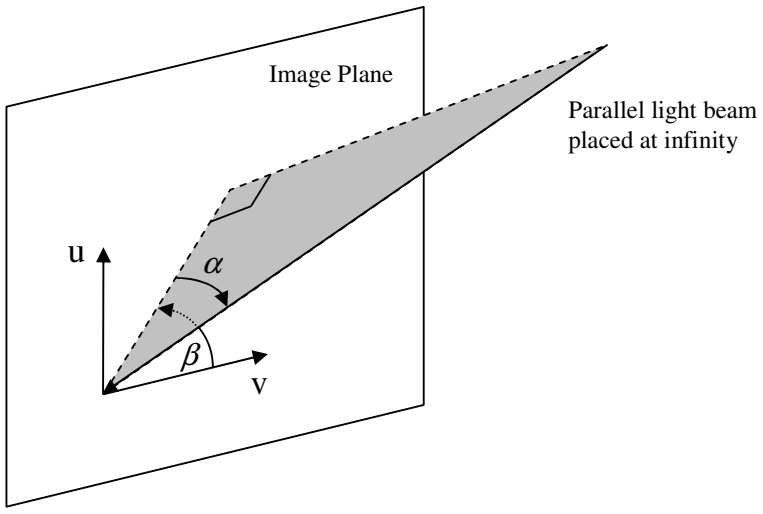
[Filename of the perspective projected intensity map]

[1] The depth at (u_i, v_j) is undefined if $D_{i,j} = 999999.0$ or $D_{i,j} = D_{u.}$

[2] After the verification of undefined depth pixel, the actual depth of $D_{i,j} \leftarrow D_B - D_{i,j}$

[3] If $M_r = 0$, the provided intensity map is rendered under parallel and uniform light beam at the infinity.

[4] The oblique angle α and β are defined as:



B. Description of major function modules

Function name	Description
SfS_Project_Load	*Load SfS project*
SfS_Project_Information	*Display the SfS project's information:* *1. Size of intensity map* *2. Focal length* *3. Rendering mode* *4. Rendering parameter* *5. Oblique angles* *6. Number of singular points*
Perspective_FMM_Frontal_Initialization	*Initialize the status and the estimated depth before performing the perspective FMM*
Perspective_FMM_Frontal_Kernel	*Core of the perspective FMM. The estimated depthmap is stored at the element* ***estDepthMap***
Perspective_FMM_Frontal_Depth_CC	*Recover the pixel's depth based on it neighboring depths*
NonBoundaryError	*Return the RMS depth error where the silhouette pixels are ignored*
DepthMap_Load	*Load the depth map with the format described in Part A to memory*
DepthMap_Write	*Write the depth map with the format described in Part A to file*
IntensityMap_Load	*Load the intensity map with the format described in Part A to memory*

IntensityMap_Write	*Write the intensity map with the format described in Part A to file*
ErrorMap_Write	*Write the absolute depth error to file in the raster scanning order*

C. Simplest way to build a perspective SfS application

Step 1: Use **SfS_Project_Load** to load the data indicated at project file to variable *cur_Model.*

Step 2: Initialized the perspective FMM
i.e. **Perspective_FMM_Frontal_Initialization**(&*cur_Model*)

Step 3: Perform the perspective FMM
i.e. **Perspective_FMM_Frontal_Kernel**(&*cur_Model*)

Step 4: Write the result *cur_Model.estDepthMap* to file using **DepthMap_Write**

Step 5: Free the memory of *cur_Model*
i.e. **SfS_Project_Destruction**(*cur_Model*)