Structure from Motion Library

12/10/2008

1. Feature Data Structure

'Structure from Motion' method needs a feature sequence in each image starting from the first image and lasting at least n images. (n is the given number). In other words, the feature data structure should contain the following information: Feature position in each image, (x_i, y_i), for i=1,2,…,n; In that case, a link-list might be an ideal choice. So the defined feature data structures are a) trajnode, b) traj, and c) trajgrp.

All this structures have prototype in file traj.h

a) trajnode

trajnode is a node in the linkwhich stores the x and y position of a feature in one image.

Necessary Attributes:

int m_nFrameNo;

integer variable, stores the index of the frame.

int m_nFeatureNo;

integer variable, stores the index of the feature in the feature space.

double m_fx;

double variable, stores the x position of the feature in the image.

double m_fy;

double variable, stores the y position of the feature in the image.

trajnode* m_pNext;

A trajnode pointer pointing to the next trajnode linked to this one. If there is no trajnode behind it then its value is NULL

b) traj

traj is a link list of the same features across multiple frames. It is a set of trajnodes. All the trajnodes in one traj should be of the same feature.

Necessary Attributes

int m_nTrajNo;

An integer stores the index of a feature.

int m_nSFNo;

An integer denotes the starting frame index of this feature. It is according to the input image index

int m_nEFNo;

An integer denotes the ending frame index of this feature. It is according to the input image index.

double RGB[3]

An 3-dimension double array stores the R,G,B average color of that feature.

trajnode* m_pTNhead;

A trajnode pointer pointing to the head/first trajnode in the linking list.

trajnode* m_pTNtail

A trajnode pointer pointing to the tail/last trajnode in the linking list.

traj* m_pNext

A traj pointer pointing to the next traj in the feature list. If there is no traj linked to it, then its value is NULL.

traj* GetInterLink(int interSF,int interEF);

This function returns a traj/feature list starting from intersSF and ending by interEF in between this struct. So the input inters should be no less than m_nSFNo, and interEF should be no bigger than m_nEFNo. It returns a new generated traj, with all the features in between interSF and interEF.

trajnode* GetNode(int index);

This function returns a trajnode pointer pointing to the trajnode of the given index.

void Release();

This function releases the whole traj.

c) trajgrp

tragrp is a link-list of traj, storing all the features and image information. It is a sequence of all the features.

Necessary Attributes:

char* pcFNameFmt;

This is a pointer to a char array storing the input images format. It is the same format with the input of function ***GetFeaturesBySIFT***.

int m_nTrajNum;

This is an integer denotes how many traj/features are in this group.

traj* m_ptrajHead;

A pointer pointing to the first/head traj of this trajgrp.

traj* m_ptrajTail;

A pointer pointing to the end/tail traj of this trajgrp.

int SFNo;

An integer stores the starting frame index of the input image sequence.

int EFNo;

An integer stores the ending frame index of the input image sequence.

int imgW;

An integer stores the width of the input image.

int imgH;

An integer store the height of the input image.

void TrajRelease();

A trajgrp release function.

int outTrajList(char* pcFName, int nMinLen = 0);

A function output all the features into a .m file for Matlab. char* pcFName is an address of a string used for the output file name. nMinLen is an integer cast a constrain on the minimum length of the feature, e.g. all the output features should be longer than nMinLen.

int outTrajCpp(char* pcFName,int nMinLen=0);

A function output all the features into a formatted data file. pcFName denotes the output file name, and nMinLen is the minimum length required to output.

int readTrajCpp(char* pcFName);

This is a function read in a file with the file name pcFName. The file should be generated by

outTrajCpp or follow the same format.

void DrawFeaturesIndex(char* inFmt="FeaturesIndex%04d.jpg", double dblScale=0.3);

This function draw the feature index on the output image. It is for debugging. The inFmt is the output image format. And dblScale is the scale of the index number in the image.

void GetFeatureColor();

This function gets the average color of each pixel and stores in RGB[3].

2. Feature detection and Structure from Motion

This part is introducing the functions that detecting the features across the images. All the functions have the prototype in file SFMLib.h

a) trajgrp* *GetFeaturesBySIFT*(char* pcFNameFmt, int nStartFrameNo, int nEndFrameNo, int MinLen =4,bool ShowMode =false);

This function generates a new trajgrp struct storing all the features detected in the image sequence by SIFT feature.

char* pcFNameFmt: the input image format, etc 'img%02d.jpg'.

int nStartFrameNo: the starting frame number of the image sequence.

int nEndFrameNo: the ending frame number of the image sequence.

int MinLen: the minimum length of the features generated, e.g. all the features should be lasting longer than MinLen.

ShowMode: a Boolean value that denotes whether to show all the intermediate result when doing the SIFT tracking. If it is true it will show all the feature correspondence in the openCV window.

b) trajgrp* GetFeaturesByKLT(char* pcFNameFmt, int nStartFrameNo, int nEndFrameNo,int nFeatures=100,int minFeatures=80,bool ShowMode=false);

This function generates a new trajgrp struct of all the features by KLT method.

char* pcFNameFmt: the input image format.

int nStartFrameNo: the starting frame number of the image sequence.

int nEndFrameNo: the ending frame number of the image sequence.

int nFeatures: the maximum number of features detected in one frame.

Int minFeatures: the minimum number of features detected in one frame.

ShowMode: a Boolean value that denotes whether to show all the intermedia result when doing the KLT tracking. If it is true then it will generate some jpg files in the current folder.

c) trajgrp* GetScreenFeatureByFrame(trajgrp* inputTrajgrp,int n_SF,int n_EF);

This is a function that gives all the features between the specific frames beginning from n_SF and ending at n_EF, and returns a new trajgrp. For example, if we want to select all the features in frame 2 and frame 5, we can put n_SF=2 and n_EF=5.

trajgrp* inputTrajgrp: the trajgrp as an input to select the features.

int n_SF: starting frames of the wanted features

int n_EF: ending frames of the wanted features.

d) void HyperFactorSFM(trajgrp* ptrajgrp,double intK[3][3],char* outFileName,int nIt=1000,int Span=3);

This function use factorization method to build the structure from the motion.

trajgrp* ptrajgrp: the input trajgrp pointer, which should contain all the features.

double intK[3][3]: the intrinsic parameters of the camera.

char* outFileName: the output file name, in which will store all the calculated 3D points and camera positions, preferably ending by .sfm for the viewer.

int nIt: iteration steps.

int Span: the number of the frames to process together and a smaller sequence.

e) int TriSBA(trajgrp* ptrajgrp,char* outFileName,double intriK[3][3],double fThreshold,int nMinTrajLen=3);

This function uses the triangulation and sparse bundle adjustment to calculate the 3D position of the feature points and estimate the camera positions.

trajgrp* ptrajgrp: the input trajgrp pointer, which should contain all the features.

char* outFileName: the output file name, in which will store all the calculated 3D points and camera positions, preferably ending by .sfm for the viewer.

double intriK[3][3]: the intrinsic parameters of the camera.

double fThreshold: this is the threshold that determines, by how much variation, then a pixel could be considered as an outlier. Mainly used for sparse bundle adjustment.

int nMinTrajLen: the minimum length of the feature needed. To achieve stable performance it is should be no less than 3.

3. Demo

a) Before you start.

Please make sure that OpenCV is properly installed in the computer. A guide could be found at http://opencv.willowgarage.com/wiki/VisualC%2B%2B

Please link the SFMLib.lib and include SFMLib.h in the code.

Please make sure that the following three files are in the same directory of the executable file or the source code:

eucsbademo.exe

blas_win32.dll

lapack_win32.dll

They are the files needed for sparse bundle adjustment and should be always with the source code/exe file.

Please make sure that the images are in the format of strings and a set of consecutive numbers, for this is the way that the program reads in the images. Some valid image sequence names could be: 'img001.jpg', 'img002.jpg', …'img005.jpg'. Then the image format is 'img%03d.jpg' and starting frame and ending frame are 1 and 5.

Please use GLpro2.4.FIFO.exe viewer to see the result.

b) In the main()

Suppose we have the image sequence as 'CH00.jpg', 'CH01.jpg', …, 'CH10.jpg'. Thus the image format is 'CH%02d.jpg' and the starting frame and ending frame are 0 and 10. We use SIFT feature detection and triangulation to reconstruct the 3D features, then the main function could be:

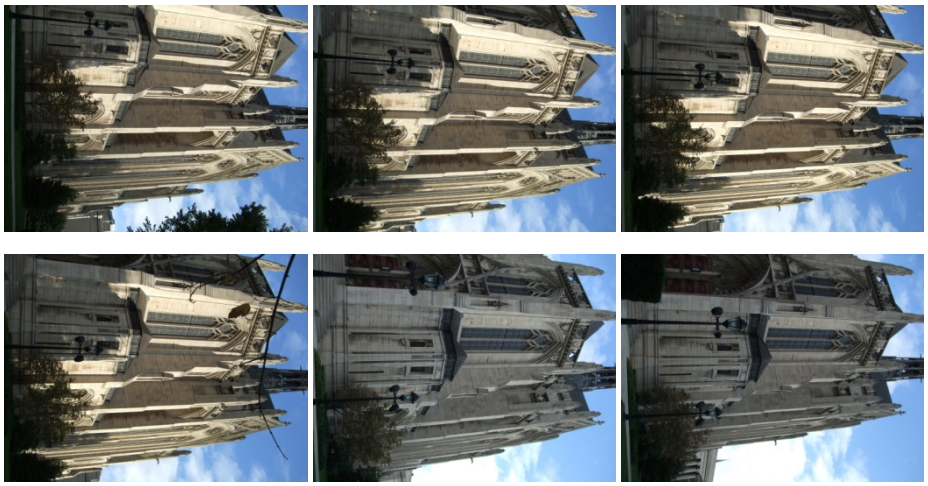
```
int main()
{

    //the intrinsic matrix K
    double intriK[3][3]={
    {2078.595725,  0.0,  963.103805},
    {0.0,   2103.087505,  760.702205},
    {0.0,  0.0,  1.0}
    };//

    trajgrp* ptrajgrp;
    int SF,EF,mLen;
    bool ShowMode;
    char* fileformat;

    mLen=3;
    ShowMode=false;
    fileformat="CH%02d.jpg";
    SF=0;
    EF=10;
    printf("Begin SIFT feature selection......\n");
    ptrajgrp=GetFeaturesBySIFT(fileformat,SF,EF,mLen,ShowMode);
    printf("End SIFT feature selection\n......");
    ptrajgrp->GetFeatureColor();
    TriSBA(ptrajgrp,"FinalResult.sfm",intriK,5);
    ptrajgrp->TrajRelease();
    system("pause");
    return 0;
}
```

The input image sequences are:



And the 3D reconstruction results should be: