OpenCV-3.0 module: render. 3D Graphics Rendering

Based on Project Ideas for GSoC 2014

25 Feb 2014
Revised: 20 Mar 2014

Arun JVS
http://web.iiit.ac.in/~arun.jvs
opencv: Fully automated 3D reconstruction

Matching Points

features2d\nonfree:
    FeatureDetector
    DescriptorExtractor
    DescriptorMatcher

Camera Parameters

re-project images to 3D

2D Images Only

2D-3D Matches

calib3d: Fundamental Matrix
    Then perform SfM

Calib3d: calibrateCamera

3D object
render: The missing link in the train-test loop

2D Images Only

Matching Points

features2d\nonfree:
FeatureDetector
DescriptorExtractor
DescriptorMatcher

calib3d: Fundamental Matrix
Then perform SfM

Camera Parameters

re-project images to 3D

2D-3D Matches

3D object

render: Take pictures of CG\Real world 3D Models
render: Dense training data for object (LINEMOD-ICCV11) and pose (Shotton-MSR) recognition algorithms

➢ Both papers make use of rendering to generate dense training data sets both for object detection as well as human pose detection.
render: More reasons...

➢ Given a 3D model, generate large number of images in different
  ➢ lighting conditions
  ➢ cameras
  ➢ poses

➢ This will provide dense training\testing data for lots of algorithms, otherwise hard to create experimentally:
  ➢ objdetect: linemod, face(cascades), body(pose), hand(ego-centric, gesture recognition), etc.
  ➢ video: motion detection, background subtraction, etc.
  ➢ features2d: finding matching points in images
  ➢ stictching: registration\alignment

➢ Generate images of both CAD and real objects (3D Scan\Kinect)

➢ Why implement in OpenCV?
  ➢ OpenCV data structures and interface
  ➢ Completes the data set requirements of all these algorithms
render: Architecture (Interface)

➢ All those details will be abstracted by opencv structures (K matrix, rvecs, tvecs)

➢ render will be an opencv module with the ModelRender class - implements 3D-mesh loading, texture loading, material properties, lighting properties, camera transformations

➢ Interface extremely similar to VideoCapture class from highgui module: **As though the video source is specified by <filename, camera-trajectory, lighting-trajectory> instead of <filename> or <device>**

```cpp
class CV_EXPORTS_W ModelRender
{
    public:
        CV_WRAP ModelRender(const String& winname);
        CV_WRAP ModelRender(const String& winname, const String& filename);

        virtual ~ModelRender();
        CV_WRAP virtual bool open(const String& filename);
        CV_WRAP virtual bool isOpened() const;
        CV_WRAP virtual void close();
        CV_WRAP virtual bool read(OutputArray image);
        virtual ModelRender& operator >> (CV_OUT Mat& image);
        virtual ModelRender& operator >> (CV_OUT UMat& image);
        CV_WRAP virtual bool set(int propId, InputArray value);
        CV_WRAP virtual bool get(int propId, OutputArray value);
};
```