Accurate, Dense, and Robust Multi-View Stereopsis

Authors:
Raúl Romaní Flores
Paul Alonzo Quio Añamuro

Source code compilation

Install some dependencies

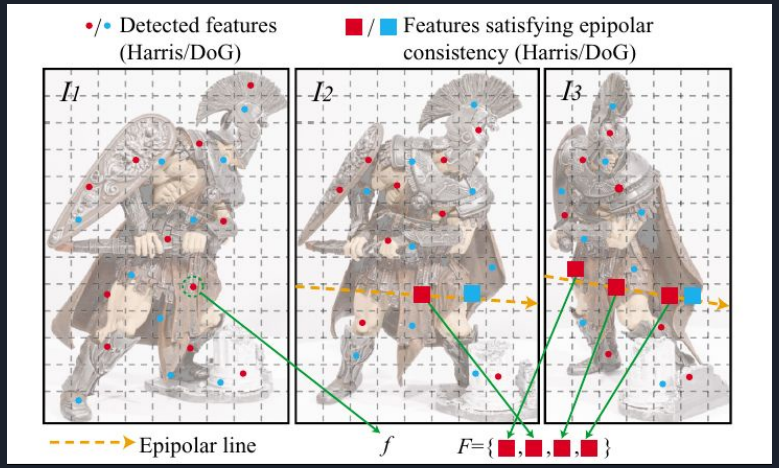
- libgtk2.0-dev
- libglew1.6-dev
- libglew1.6
- libdevil-dev
- libboost-all-dev
- libatlas-cpp-0.6-dev
- libatlas-dev
- imagemagick
- libatlas3gf-base
- libcminpack-dev
- libgfortran3
- libmetis-edf-dev
- libparmetis-dev
- freeglut3-dev
- libgsl0-dev
- libblas-dev
- liblapack-dev
- liblapacke-dev
- libjpeg-dev

And use make to generate executable files

```
alonzo@alonzo-VirtualBox:~/Documents/program/main$ make
g++ -O2 -Wall -Wno-deprecated    -c -o pmvs2.o pmvs2.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/detectFeatures.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/dog.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/harris.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/point.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/detector.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/findMatch.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/expand.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/filter.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/optim.cc
../base/pmvs/optim.cc: In static member function ‘static double PMVS3::Coptim::my
_f_ssd(const gsl_vector*, void*)’:
../base/pmvs/optim.cc:762:9: warning: variable ‘flag’ set but not used [-Wunused-
but-set-variable]
     int flag;
         ^
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/patchOrganizerS.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/seed.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/option.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/image/image.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/image/camera.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/image/photoSetS.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/pmvs/patch.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/image/photo.cc
g++ -c -O2 -Wall -Wno-deprecated  ../base/numeric/mylapack.cc
g++  -lXext -lX11 -ljpeg -lm -lpthread -llapack -lgsl -lgslcblas -o pmvs2 pmvs2.o
 detectFeatures.o dog.o harris.o point.o detector.o findMatch.o expand.o filter.o
 optim.o patchOrganizerS.o seed.o option.o image.o camera.o photoSetS.o patch.o p
hoto.o mylapack.o  -lXext -lX11 -ljpeg -lm -lpthread -llapack -lgsl -lgslcblas
alonzo@alonzo-VirtualBox:~/Documents/program/main$ ls
camera.o          findMatch.o       liblapack.so.3  patchOrganizerS.o  point.o
detectFeatures.o  harris.o          Makefile        photo.o            run0.sh
detector.o        image.o           mylapack.o      photoSetS.o        run1.sh
dog.o             libblas.so.3      optim.o         pmvs2              run2.sh
expand.o          libgslcblas.so.0  option.o        pmvs2.cc           seed.o
filter.o          libgsl.so.0       patch.o         pmvs2.o
alonzo@alonzo-VirtualBox:~/Documents/program/main$
```

Code - paper matching

Detect corner and blob features in each image using the Harris and Difference of-Gaussian (DoG) operators.




```
// Harris
{
    Charris harris;
    multiset<Cpoint> result;
    harris.run(m_ppss->m_photos[index].getImage(m_level),
               m_ppss->m_photos[index].Cimage::getMask(m_level),
               m_ppss->m_photos[index].Cimage::getEdge(m_level),
               m_ppss->m_photos[index].getWidth(m_level),
               m_ppss->m_photos[index].getHeight(m_level), m_csize, sigma, result);

    multiset<Cpoint>::reverse_iterator rbegin = result.rbegin();
    while (rbegin != result.rend()) {
        m_points[index].push_back(*rbegin);
        rbegin++;
    }
}

//---------------------------------------------------------------------
// DoG
{
    Cdog dog;
    multiset<Cpoint> result;
    dog.run(m_ppss->m_photos[index].getImage(m_level),
            m_ppss->m_photos[index].Cimage::getMask(m_level),
            m_ppss->m_photos[index].Cimage::getEdge(m_level),
            m_ppss->m_photos[index].getWidth(m_level),
            m_ppss->m_photos[index].getHeight(m_level),
            m_csize, firstScale, lastScale, result);

    multiset<Cpoint>::reverse_iterator rbegin = result.rbegin();
    while (rbegin != result.rend()) {
        m_points[index].push_back(*rbegin);
        rbegin++;
    }
}
```

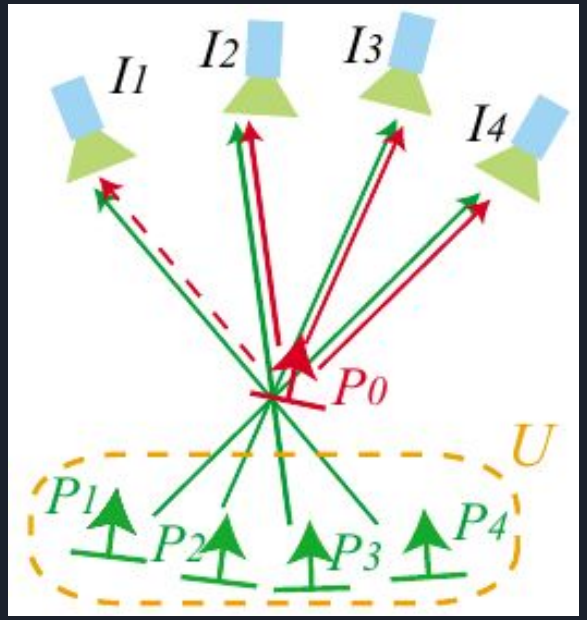
Expansion

```
void Cexpand::run(void) {
    m_fm.m_count = 0;
    m_fm.m_jobs.clear();
    m_ecounts.resize(m_fm.m_CPU);
    m_fcounts0.resize(m_fm.m_CPU);
    m_fcounts1.resize(m_fm.m_CPU);
    m_pcounts.resize(m_fm.m_CPU);
    fill(m_ecounts.begin(), m_ecounts.end(), 0);
    fill(m_fcounts0.begin(), m_fcounts0.end(), 0);
    fill(m_fcounts1.begin(), m_fcounts1.end(), 0);
    fill(m_pcounts.begin(), m_pcounts.end(), 0);

    time_t starttime = time(NULL);

    m_fm.m_pos.clearCounts();
    m_fm.m_pos.clearFlags();

    if (!m_queue.empty()) {
        cerr << "Queue is not empty in expand" << endl;
        exit (1);
    }
    // set queue
    m_fm.m_pos.collectPatches(m_queue);

    cerr << "Expanding patches..." << flush;
    pthread_t threads[m_fm.m_CPU];
    for (int c = 0; c < m_fm.m_CPU; ++c)
        pthread_create(&threads[c], NULL, expandThreadTmp, (void*)this);
    for (int c = 0; c < m_fm.m_CPU; ++c)
        pthread_join(threads[c], NULL);

    cerr << endl
         << "---- EXPANSION: " << (time(NULL) - starttime) << " secs ----" << endl;

    const int trial = accumulate(m_ecounts.begin(), m_ecounts.end(), 0);
    const int fail0 = accumulate(m_fcounts0.begin(), m_fcounts0.end(), 0);
    const int fail1 = accumulate(m_fcounts1.begin(), m_fcounts1.end(), 0);
    const int pass = accumulate(m_pcounts.begin(), m_pcounts.end(), 0);
```

Filter

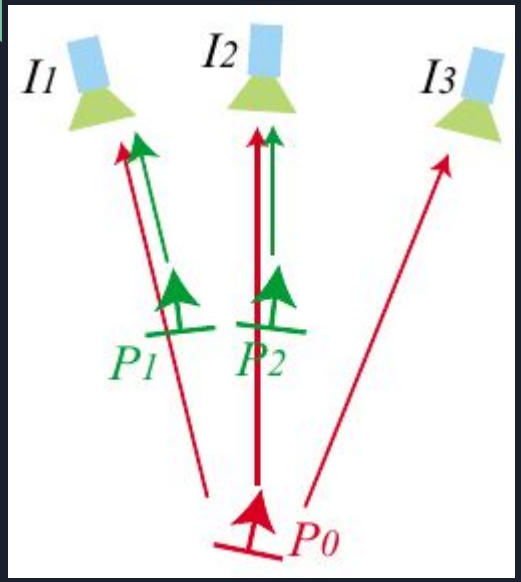
```
void Cfilter::run(void) {
    setDepthMapsVGridsVPGridsAddPatchV(0);

    filterOutside();
    setDepthMapsVGridsVPGridsAddPatchV(1);

    filterExact();
    setDepthMapsVGridsVPGridsAddPatchV(1);

    filterNeighbor(1);
    setDepthMapsVGridsVPGridsAddPatchV(1);

    filterSmallGroups();
    setDepthMapsVGridsVPGridsAddPatchV(1);
}
```

Filter outside




```
void Cfilter::filterOutside(void) {
    struct timeval tv;
    gettimeofday(&tv, NULL);
    time_t curtime = tv.tv_sec;
    cerr << "FilterOutside" << endl;
    //??? notice (1) here to avoid removing m_fix=1
    m_fm.m_pos.collectPatches(1);

    const int psize = (int)m_fm.m_pos.m_ppatches.size();
    m_gains.resize(psize);

    cerr << "mainbody: " << flush;

    m_fm.m_count = 0;
    pthread_t threads[m_fm.m_CPU];
    for (int i = 0; i < m_fm.m_CPU; ++i)
        pthread_create(&threads[i], NULL, filterOutsideThreadTmp, (void*)this);
    for (int i = 0; i < m_fm.m_CPU; ++i)
        pthread_join(threads[i], NULL);
    cerr << endl;

    // delete patches with positive m_gains
    int count = 0;

    double ave = 0.0f;
    double ave2 = 0.0f;
    int denom = 0;

    for (int p = 0; p < psize; ++p) {
        ave += m_gains[p];
        ave2 += m_gains[p] * m_gains[p];
        ++denom;

        if (m_gains[p] < 0.0) {
            m_fm.m_pos.removePatch(m_fm.m_pos.m_ppatches[p]);
            count++;
        }
    }

    if (denom == 0)
        denom = 1;
    ave /= denom;
    ave2 /= denom;
    ave2 = sqrt(max(0.0, ave2 - ave * ave));
```

Filter exact

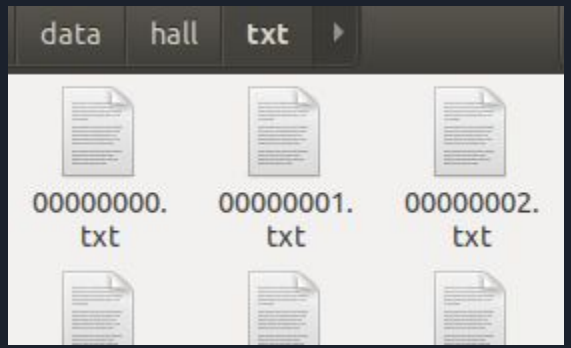


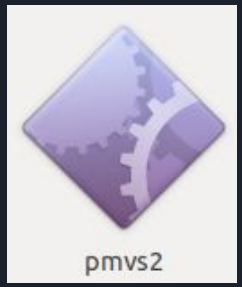

```
void Cfilter::filterExact(void) {
    struct timeval tv;
    gettimeofday(&tv, NULL);
    time_t curtime = tv.tv_sec;
    cerr << "Filter Exact: " << flush;

    //??? cannot use (1) because we use patch.m_id to set newimages,....
    m_fm.m_pos.collectPatches();
    const int psize = (int)m_fm.m_pos.m_ppatches.size();

    // dis associate images
    m_newimages.clear();     m_newgrids.clear();
    m_removeimages.clear();  m_removegrids.clear();
    m_newimages.resize(psize);     m_newgrids.resize(psize);
    m_removeimages.resize(psize);  m_removegrids.resize(psize);

    m_fm.m_count = 0;
    pthread_t threads0[m_fm.m_CPU];
    for (int i = 0; i < m_fm.m_CPU; ++i)
        pthread_create(&threads0[i], NULL, filterExactThreadTmp, (void*)this);
    for (int i = 0; i < m_fm.m_CPU; ++i)
        pthread_join(threads0[i], NULL);
    cerr << endl;

    //-------------------------------------------------------------------------
    for (int p = 0; p < psize; ++p) {
        if (m_fm.m_pos.m_ppatches[p]->m_fix)
            continue;

        for (int i = 0; i < (int)m_removeimages[p].size(); ++i) {
            const int index = m_removeimages[p][i];
            if (m_fm.m_tnum <= index) {
                cerr << "MUST NOT COME HERE" << endl;        exit (1);
            }
            const int ix = m_removegrids[p][i][0];      const int iy = m_removegrids[p][i][1];
            const int index2 = iy * m_fm.m_pos.m_gwidths[index] + ix;

            m_fm.m_pos.m_pgrids[index][index2].
            erase(remove(m_fm.m_pos.m_pgrids[index][index2].begin(),
                         m_fm.m_pos.m_pgrids[index][index2].end(),
                         m_fm.m_pos.m_ppatches[p]),
                  m_fm.m_pos.m_pgrids[index][index2].end());
        }
    }

    m_fm.m_debug = 1;
```

Demo execution

```
alonzo@alonzo-desktop:~/Documentos/Projects/pmvs-2/program/main$
./pmvs2 "/home/alonzo/Documentos/Projects/pmvs-2/data/hall/" opti
on.txt

./pmvs2
/home/alonzo/Documentos/Projects/pmvs-2/data/hall/
option.txt----------------------------------------------
--- Summary of specified options ---
# of timages: 61 (range specification)
# of oimages: 0 (enumeration)
level: 2  csize: 2
threshold: 0.7  wsize: 7
minImageNum: 3  CPU: 4
useVisData: 1  sequence: -1
-----------------------------------------------
Reading images: **********
```

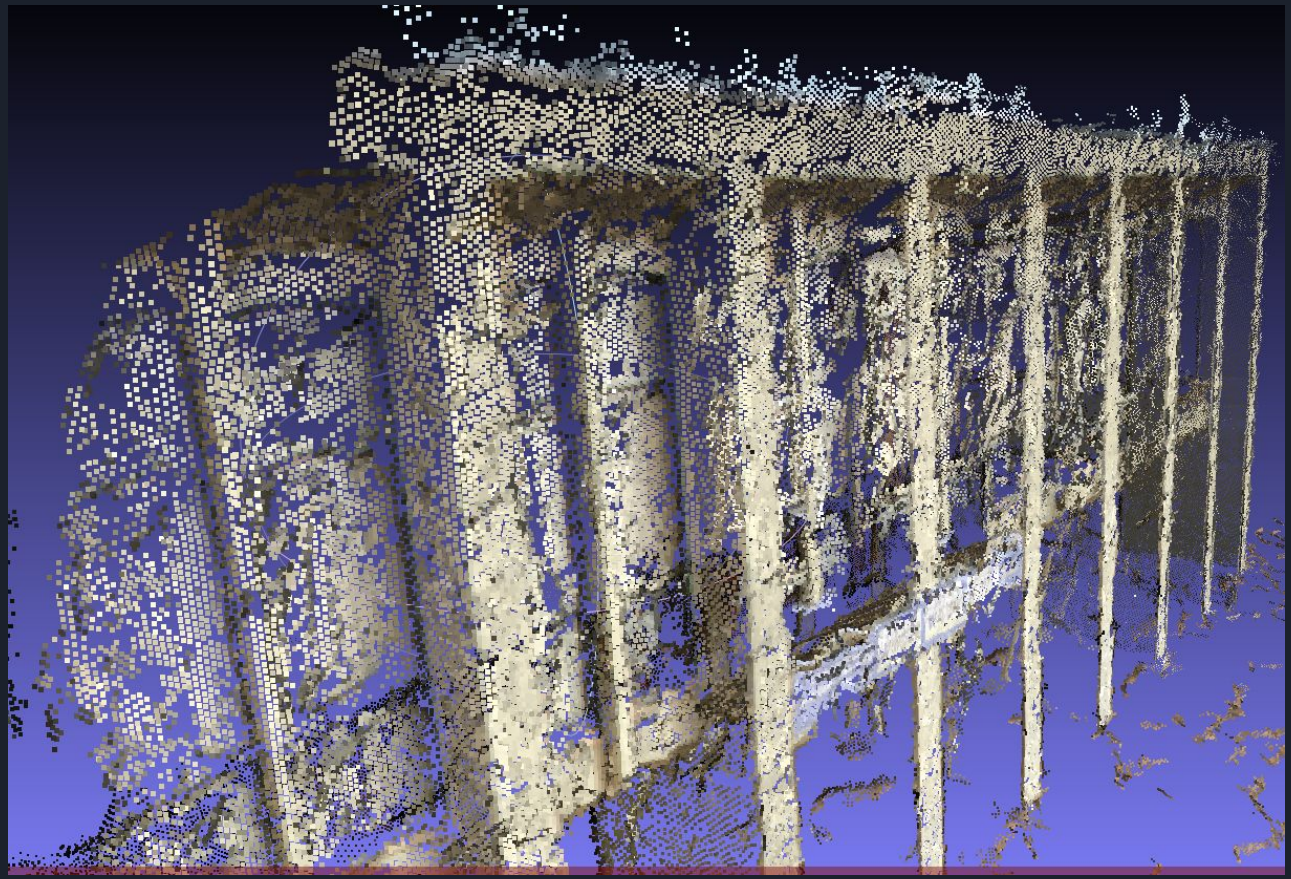
Demo execution





Results

Results - Meshlab



Thanks