Lab 5
Pose estimation and Augmented Reality

23.02.2017

Bildebasert navigasjon og AR

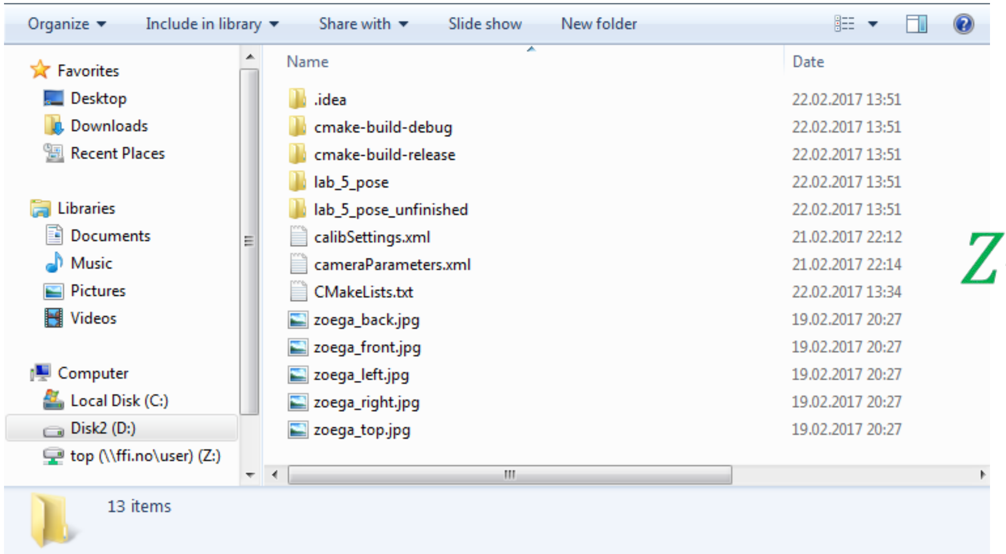
Hva trengs for å bruke et kamera til å navigere rundt i en kjent 3D verden?

- En 3D verdensmodell som vi automatisk kan detektere i bilder
- En metode som automatisk etablerer 3D-2D korrespondanser mellom verdensmodell og bilder
- En metode som estimerer kameraets pose fra kjente 3D-2D korrespondanser
  - Altså en PnP-metode

Lab 5 i grove trekk

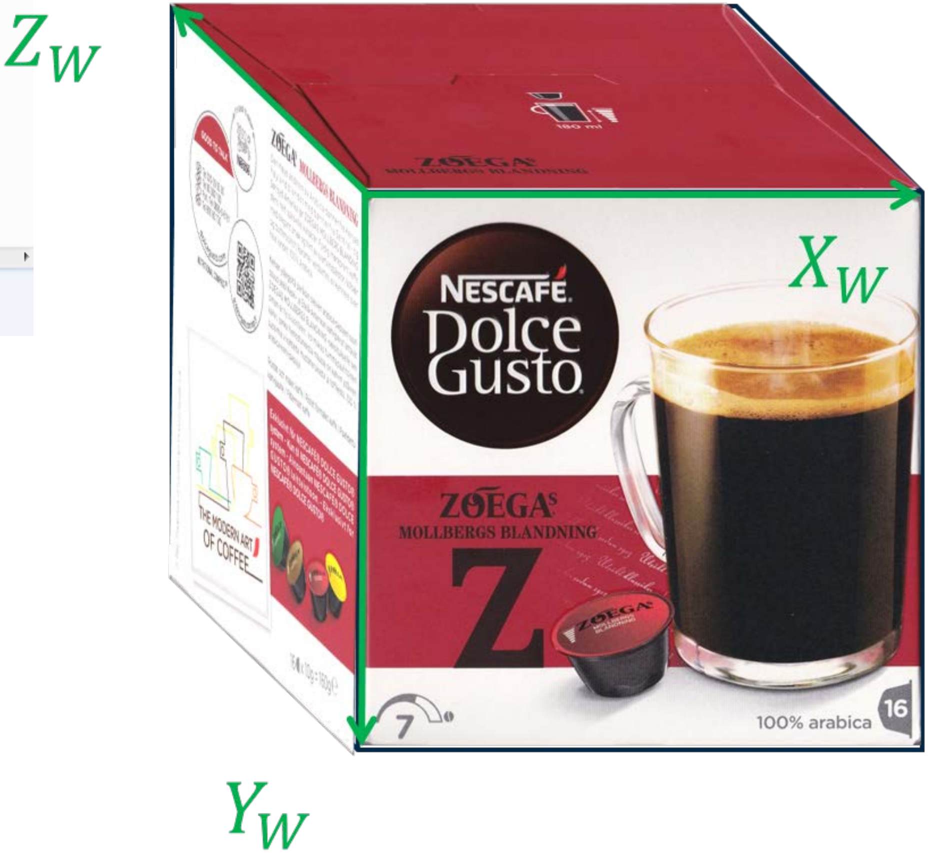
- Lag en 3D verdensmodell som automatisk kan detekteres i bilder
- Etabler 3D-2D korrespondanser mellom verdensmodell og bilder
- Estimer kamera pose fra 3D-2D korrespondansene
  - Løs PnP problemet
- 2D visualisering av verdensmodell i bildene (Augmented Reality)
- 3D visualisering av kamera og verdensmodell

Verdensmodellen



Name	Date
.idea	22.02.2017 13:51
cmake-build-debug	22.02.2017 13:51
cmake-build-release	22.02.2017 13:51
lab_5_pose	22.02.2017 13:51
lab_5_pose_unfinished	22.02.2017 13:51
calibSettings.xml	21.02.2017 22:12
cameraParameters.xml	21.02.2017 22:14
CMakeLists.txt	22.02.2017 13:34
zoega_back.jpg	19.02.2017 20:27
zoega_front.jpg	19.02.2017 20:27
zoega_left.jpg	19.02.2017 20:27
zoega_right.jpg	19.02.2017 20:27
zoega_top.jpg	19.02.2017 20:27

13 items

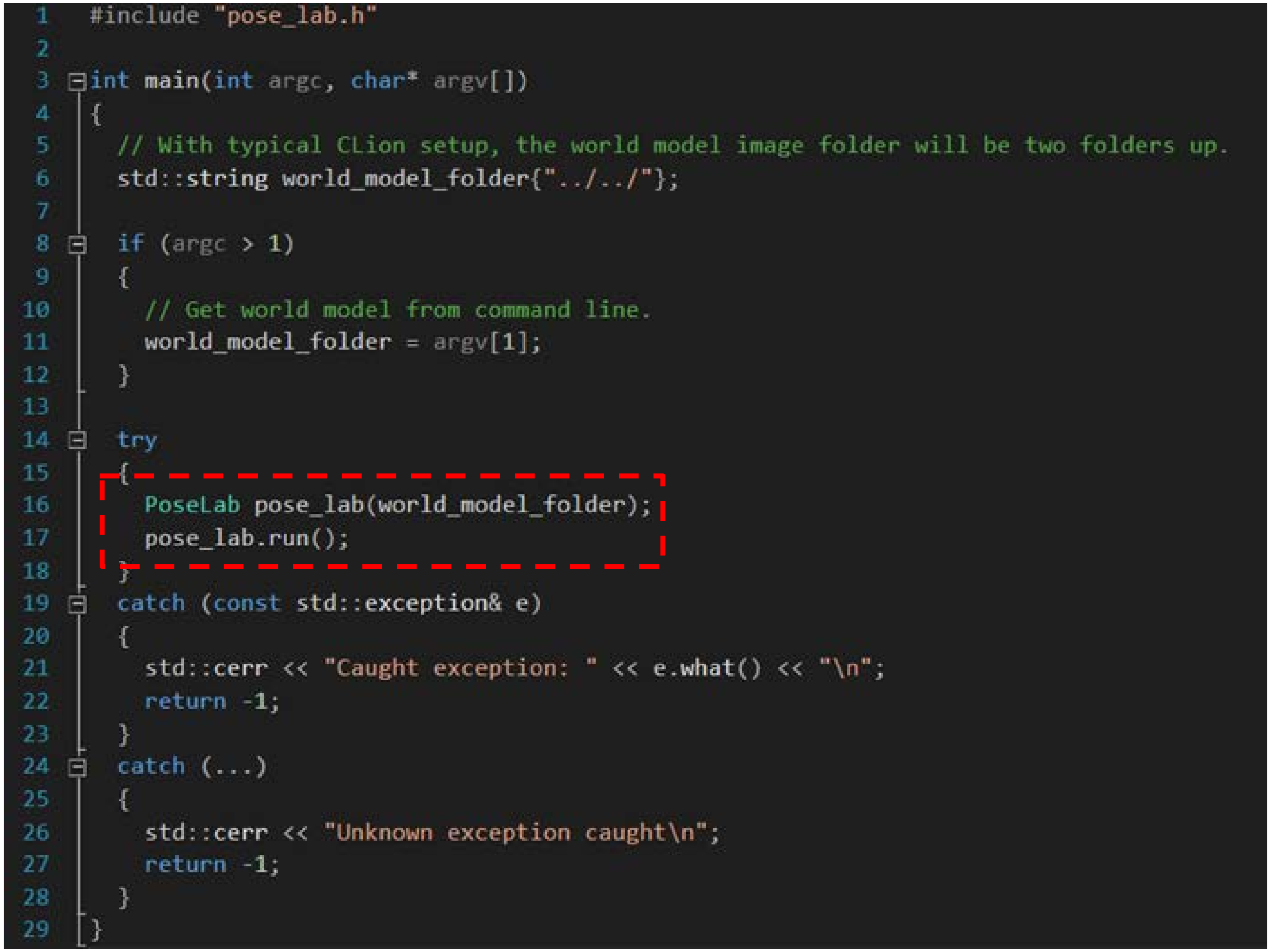


2D features med 3D posisjon

Klassene i prosjektet

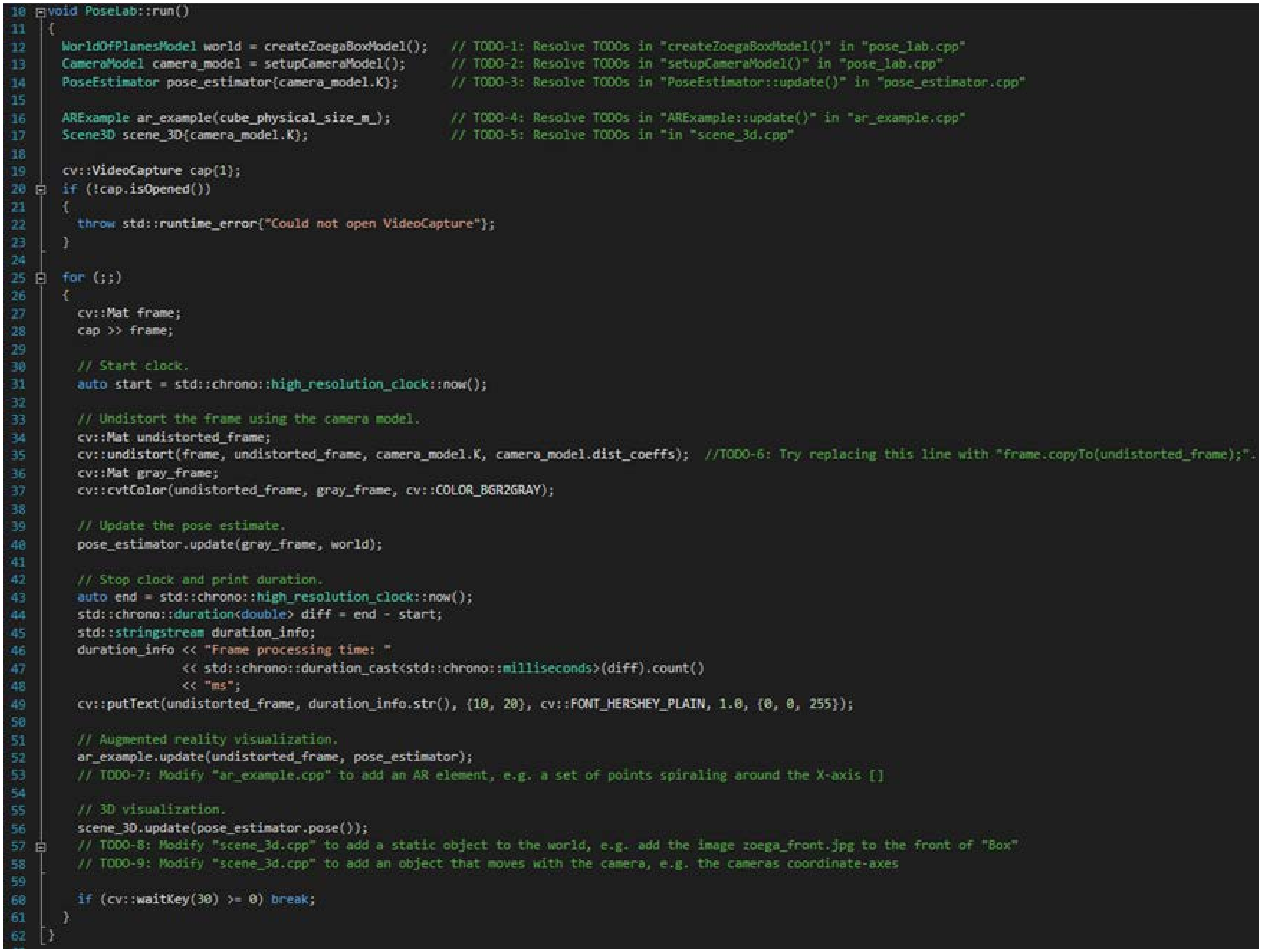
- **Pose_lab** – pose_lab.h , pose_lab.cpp
  - Hovedklassen som «kjører» prosjektet
- **WorldOfPlanesModel** – world_of_planes_model.h , world_of_planes_model.cpp
  - Klasse for å administrerer verdensmodellen som bygges opp av plan
- **PlaneReference** – plane_reference.h
  - Klasse for å administrere enkelt plan
- **PoseEstimator** – pose_estimator.h , pose_estimator.cpp
  - Klasse for å administrere pose estimeringen
- **ARExample** – ar_example.h , ar_example.cpp
  - Klasse for å administrere 2D visningen, dvs live-bilder med AR
- **Scene3D** – scene_3d.h , scene_3d.cpp
  - Klasse for å administrere 3D visningen, dvs kamera i verden

Main funksjonen



```
#include "pose_lab.h"

int main(int argc, char* argv[])
{
  // With typical CLion setup, the world model image folder will be two folders up.
  std::string world_model_folder{"../../"};

  if (argc > 1)
  {
    // Get world model from command line.
    world_model_folder = argv[1];
  }

  try
  {
    PoseLab pose_lab(world_model_folder);
    pose_lab.run();
  }
  catch (const std::exception& e)
  {
    std::cerr << "Caught exception: " << e.what() << "\n";
    return -1;
  }
  catch (...)
  {
    std::cerr << "Unknown exception caught\n";
    return -1;
  }
}
```

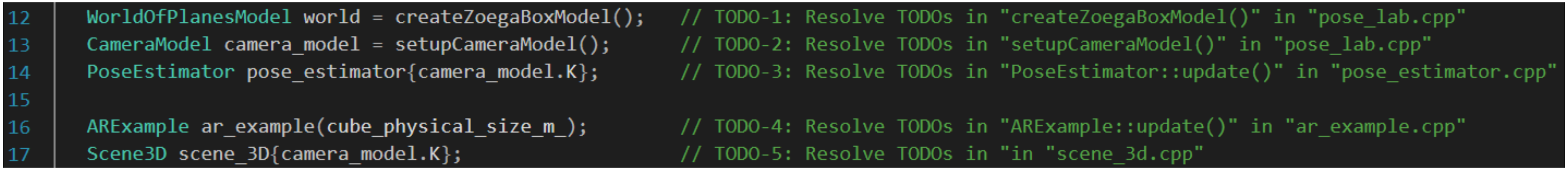
PoseLab.run()



```cpp
void PoseLab::run()
{
  WorldOfPlanesModel world = createZoegaBoxModel();   // TODO-1: Resolve TODOs in "createZoegaBoxModel()" in "pose_lab.cpp"
  CameraModel camera_model = setupCameraModel();      // TODO-2: Resolve TODOs in "setupCameraModel()" in "pose_lab.cpp"
  PoseEstimator pose_estimator{camera_model.K};       // TODO-3: Resolve TODOs in "PoseEstimator::update()" in "pose_estimator.cpp"

  ARExample ar_example(cube_physical_size_m_);        // TODO-4: Resolve TODOs in "ARExample::update()" in "ar_example.cpp"
  Scene3D scene_3D{camera_model.K};                   // TODO-5: Resolve TODOs in "in "scene_3d.cpp"

  cv::VideoCapture cap{1};
  if (!cap.isOpened())
  {
    throw std::runtime_error{"Could not open VideoCapture"};
  }

  for (;;)
  {
    cv::Mat frame;
    cap >> frame;

    // Start clock.
    auto start = std::chrono::high_resolution_clock::now();

    // Undistort the frame using the camera model.
    cv::Mat undistorted_frame;
    cv::undistort(frame, undistorted_frame, camera_model.K, camera_model.dist_coeffs);  //TODO-6: Try replacing this line with "frame.copyTo(undistorted_frame);".
    cv::Mat gray_frame;
    cv::cvtColor(undistorted_frame, gray_frame, cv::COLOR_BGR2GRAY);

    // Update the pose estimate.
    pose_estimator.update(gray_frame, world);

    // Stop clock and print duration.
    auto end = std::chrono::high_resolution_clock::now();
    std::chrono::duration<double> diff = end - start;
    std::stringstream duration_info;
    duration_info << "Frame processing time: "
                  << std::chrono::duration_cast<std::chrono::milliseconds>(diff).count()
                  << "ms";
    cv::putText(undistorted_frame, duration_info.str(), {10, 20}, cv::FONT_HERSHEY_PLAIN, 1.0, {0, 0, 255});

    // Augmented reality visualization.
    ar_example.update(undistorted_frame, pose_estimator);
    // TODO-7: Modify "ar_example.cpp" to add an AR element, e.g. a set of points spiraling around the X-axis []

    // 3D visualization.
    scene_3D.update(pose_estimator.pose());
    // TODO-8: Modify "scene_3d.cpp" to add a static object to the world, e.g. add the image zoega_front.jpg to the front of "Box"
    // TODO-9: Modify "scene_3d.cpp" to add an object that moves with the camera, e.g. the cameras coordinate-axes

    if (cv::waitKey(30) >= 0) break;
  }
}
```

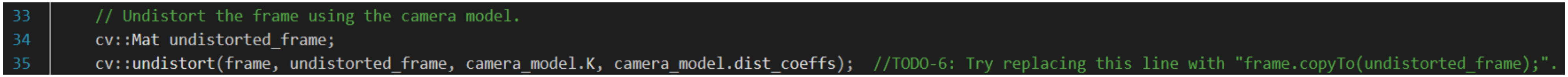
PoseLab.run() – Del 1

- Forstå PoseLab og gjør den bedre



```
WorldOfPlanesModel world = createZoegaBoxModel();   // TODO-1: Resolve TODOs in "createZoegaBoxModel()" in "pose_lab.cpp"
CameraModel camera_model = setupCameraModel();      // TODO-2: Resolve TODOs in "setupCameraModel()" in "pose_lab.cpp"
PoseEstimator pose_estimator{camera_model.K};       // TODO-3: Resolve TODOs in "PoseEstimator::update()" in "pose_estimator.cpp"

ARExample ar_example(cube_physical_size_m_);        // TODO-4: Resolve TODOs in "ARExample::update()" in "ar_example.cpp"
Scene3D scene_3D{camera_model.K};                   // TODO-5: Resolve TODOs in "in "scene_3d.cpp"
```

- Hva gjør undistortion her?



```
// Undistort the frame using the camera model.
cv::Mat undistorted_frame;
cv::undistort(frame, undistorted_frame, camera_model.K, camera_model.dist_coeffs);  //TODO-6: Try replacing this line with "frame.copyTo(undistorted_frame);".
```

- Har det noen effekt på pose estimeringen om vi fjerner denne «undistorten»?
  - Hvorfor/hvorfor ikke?

TODO-2-1: Kamera kalibrering

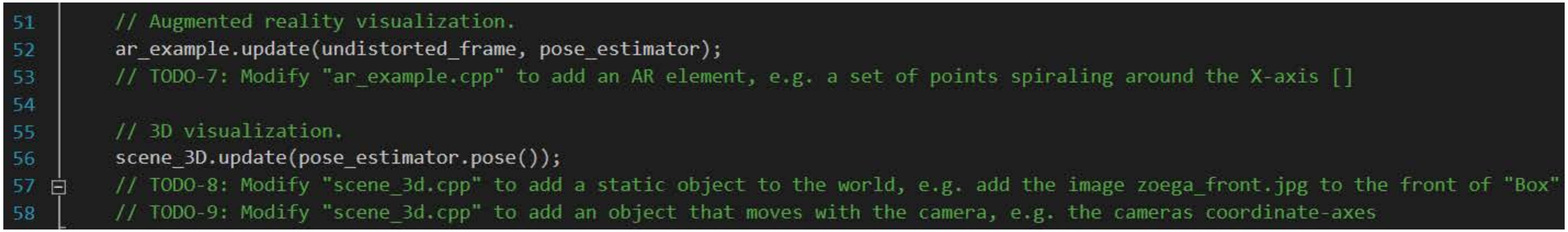
- OpenCV appen opencv_interactive-calibration gjør det enkelt for oss å kalibrere kameraene
  - Legg et sjakkbrett flatt på pulten
  - Åpne en konsoll `Ctrl + alt + t`
  - Gå til prosjekt katalogen, f.eks. `cd/projects/lab_5`
  - Kjør opencv_interactive-calibration med kommandoen

```
opencv_interactive-calibration -ci=1 –t=chessboard –sz=30 –w=7 –h=5 –pf=calibSettings.xml
```

  - Når du er fornøyd, lagrer du resultatet til filen cameraParameters.xml ved å trykke på `s`
  - Appen avsluttes ved å trykke på `Esc`

PoseLab.run() – Del 2

- Utvid PoseLab med egne kule greier



```
    // Augmented reality visualization.
    ar_example.update(undistorted_frame, pose_estimator);
    // TODO-7: Modify "ar_example.cpp" to add an AR element, e.g. a set of points spiraling around the X-axis []

    // 3D visualization.
    scene_3D.update(pose_estimator.pose());
    // TODO-8: Modify "scene_3d.cpp" to add a static object to the world, e.g. add the image zoega_front.jpg to the front of "Box"
    // TODO-9: Modify "scene_3d.cpp" to add an object that moves with the camera, e.g. the cameras coordinate-axes
```