

Respiratory Signal Using Trajectory Triangulation (December 2007)

The files contained in this directory are the matlab source code for respiratory signal generation using a spherical skin fiducial imaged with a cone-beam CT¹. If you use the code in your research please cite it as:

S. Wiesner, Z. Yaniv, "Respiratory Signal Generation for Retrospective Gating of Cone-Beam CT Images", *SPIE Medical Imaging: Visualization, Image-Guided Procedures, and Display*, 2008.

NOTE: The complete code base uses both an analytic and a nonlinear method to perform trajectory estimation. The nonlinear method for trajectory estimation requires matlab's optimization toolbox. The code will not generate an error if the toolbox is missing. Rather, we chose to check for this situation (see 'haveToolbox' function) and silently skip over all the functions that require the nonlinear optimization toolbox.

Algorithmically interesting functions:

- analyticTrajectoryEstimation.m
- nonlinearTrajectoryEstimation.m

Simulation scripts:

- runSimulation.m - runs the following three functions.
- runRandomTrajectories.m - Generate random linear trajectories close to the y axis, angular deviation from y axis is constrained to be up to 45 degrees. Simulated respiratory data is created for each of the trajectories and the analytic and nonlinear methods for trajectory estimation are evaluated. Quantitative output is written to a text file and a plot of the random trajectories ('randomTrajectories.eps,emf,fig') is saved.
- runTrajectoryWithNoise.m - Compare the analytic and nonlinear trajectory estimation methods using multiple runs to estimate a known trajectory $[p,n] = [0,0,0,0,1/\sqrt{2},1/\sqrt{2}]$, using simulated respiration with noise added to the 2D points. Quantitative output is written to a text file.
- runTrajectoryWithRespiration.m - Compare the analytic and nonlinear trajectory estimation methods using tracking data acquired in the thoracic abdominal region of a volunteer. The trajectories are then used to generate a respiratory signal that is visually compared to the respiratory signal generated directly from the 3D tracking data. Quantitative output is written to a text file and a plot of all the estimated trajectories and the corresponding 3D tracking data is saved. In addition all trajectories are saved in individual plots. Finally, respiratory plots are saved for each of the fiducials.

¹Code runs successfully on matlab version 7.4.0.287 with optimization toolbox version 3.1.1.

Data files:

- projectionMatrices.txt - ASCII file containing the C-arm's calibration matrices.
- optotrakData.txt - ASCII file containing tracking data from optical markers placed in the thoracic-abdominal region of a volunteer (see optotrakDataAcquisition.jpg for fiducial locations). Data was acquired with the Optotrak Certus system (Northern Digital Inc.) using an acquisition rate of 60Hz. Only seven of the 12 markers were tracked, due to line of sight issues arising from the constraints on positioning the optotrak camera system in our laboratory.

To run the simulation just type `runSimulation`. All the plots and a text file with the quantitative results are written to the 'results' directory. Make sure that this directory exists before running the script.

For questions and/or bug reports contact Ziv Yaniv, zivy@isis.georgetown.edu

Distribution manifest:

1. README.pdf
2. analyticTrajectoryEstimation.m
3. approximationError.m
4. createAxisAngleTransformation.m
5. createBackprojectedRaySet.m
6. crossMatrix.m
7. generateRespiratoryDataFromVolunteer.m
8. generateSimulatedRespiratoryData.m
9. haveToolbox.m
10. leastSquaresRayIntersection.m
11. lineLineMinDistance.m
12. loadOptotrakPoints.m
13. loadProjectionMatrices.m
14. nonlinearTrajectoryEstimation.m
15. optotrakData.txt
16. optotrakDataAcquisition.jpg
17. pca.m
18. plucker2PointDirectionLineParameterization.m
19. pointProjectionMatrix2PluckerLineProjectionMatrix.m
20. projectionMatrices.txt
21. respiratorySignalFrom4DPoint.m
22. runRandomTrajectories.m
23. runSimulation.m
24. runTrajectoryWithNoise.m
25. runTrajectoryWithRespiration.m
26. saveFigure.m
27. trajectoryComparison.m
28. triangulateTrajectory.m

```

/*****
* Copyright (c) 2007
* Computer Aided Interventions and Medical Robotics (CAIMR) group,
* Imaging Science and Information Systems (ISIS) research center,
* Georgetown University.
*
* All rights reserved.
*
* Permission to use, copy, modify, distribute, and sell this software
* and its documentation for any purpose is hereby granted without fee,
* provided that:
* (1) Redistributions of source code must retain the above copyright notice,
*     this list of conditions and the following disclaimer.
* (2) Redistributions in binary form must reproduce the above copyright notice,
*     this list of conditions and the following disclaimer in the documentation
*     and/or other materials provided with the distribution.
* (3) The author's name may not be used to endorse or promote products derived
*     from this software without specific prior written permission.
* (4) Modified source versions must be plainly marked as such, and must not be
*     misrepresented as being the original software.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER 'AS IS' AND ANY EXPRESS
* OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
* IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
* PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
* OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
* WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
* OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
*****/
```