

October 2007

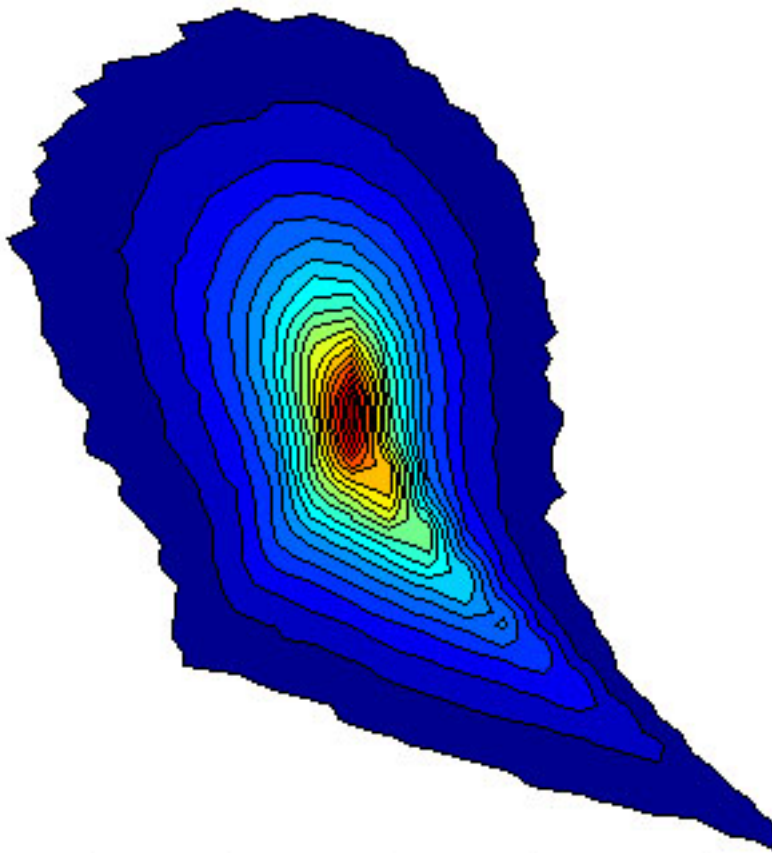
The principal of the following software is described in pages 6-9 of Lüthi et al., JFM 2005 ([Luthi_2005_JFM_deriv_PTV.pdf](#)) and the core for the derivatives is described on page 3 of Lüthi et al., JoT 2007 ([Luthi_2007_JoT_multi_part.pdf](#)).

Download here:

We now have a no-GUI, very up-to-date, and reduced-to-the-max version of [post_process.exe](#) post_process software. It is written in C, MVS, has one single ASCII input file ([input.inp](#)) and as output has ASCII files with filtered:

x,y,z, u,v,w ax,ay,az, du/dt (3) du/dx (9) da/dx (9)

It can be used to make, e.g., RQ plots from ptv_is files:



The source code is in [post_process.cpp](#) and [stdafx.h](#).

How to run it?

save the two files *post_process.exe* and your edited *input.inp* in a folder and go there in a command console.
Type *post_process input.inp* and press enter. Now it should run and it should look like:

```
C:\WINDOWS\system32\cmd.exe - post_process input.inp
D:\>cd transport
D:\transport>dir
Volume in drive D is Data
Volume Serial Number is E0D3-E806

Directory of D:\transport

09.10.2007  11:05    <DIR>          .
09.10.2007  11:05    <DIR>          ..
09.10.2007  10:37             1'078 input.inp
09.10.2007  10:48        100'118 input_ASCII_file.jpg
09.10.2007  10:37             1'078 post_proc.inp
09.10.2007  10:45        61'138 post_process.cpp
09.10.2007  10:50        40'960 post_process.exe
09.10.2007  10:45       89'985 screen_ouput.jpg
09.10.2007  10:28           2'488 stdafx.h
09.10.2007  10:19       153'835 trajPoint_ASCII_format.jpg
               8 File(s)              450'680 bytes
               2 Dir(s)  14'032'412'672 bytes free

D:\transport>post_process input.inp

succesfully opened *.inp file
processing file .....10000
max Vel.....0
mean Vel.....0
mean Acc.....0

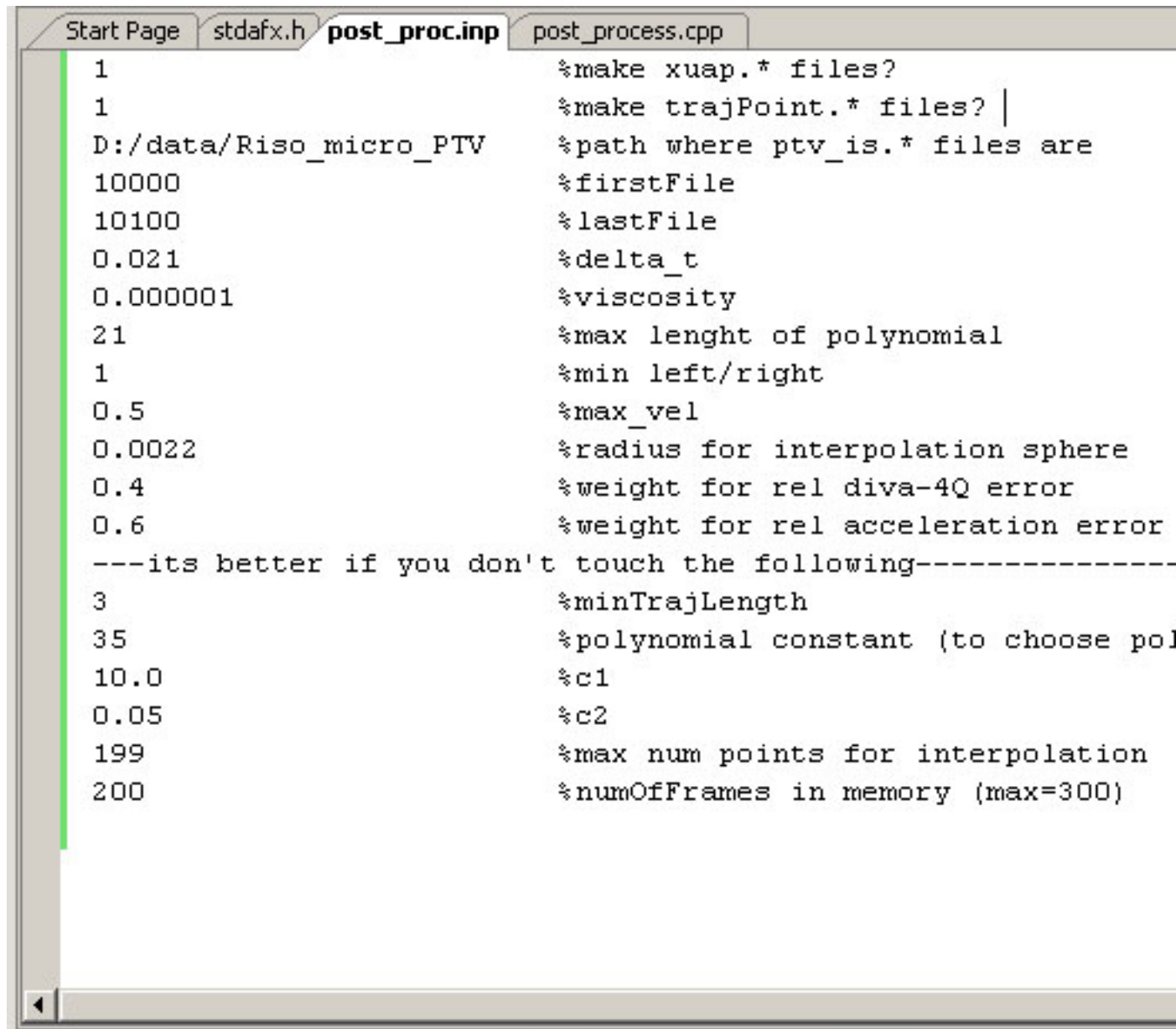
processing file .....10020
max Vel.....0.0459165
mean Vel.....0.0146403
mean Acc.....0.0322564

processing file .....10040
max Vel.....0.0579238
mean Vel.....0.0141533
mean Acc.....0.0309714

processing file .....10060
max Vel.....0.0579238
mean Vel.....0.0137557
mean Acc.....0.0302794
```

How to control it?

In the input file you define essentially 13 parameters.



```
1 %make xuap.* files?
1 %make trajPoint.* files? |
D:/data/Riso_micro_PTV %path where ptv_is.* files are
10000 %firstFile
10100 %lastFile
0.021 %delta_t
0.000001 %viscosity
21 %max lenght of polynomial
1 %min left/right
0.5 %max_vel
0.0022 %radius for interpolation sphere
0.4 %weight for rel diva-4Q error
0.6 %weight for rel acceleration error
---its better if you don't touch the following-----
3 %minTrajLength
35 %polynomial constant (to choose pol
10.0 %c1
0.05 %c2
199 %max num points for interpolation
200 %numOfFrames in memory (max=300)
```

The first two flags determine which parts should be executed: ptv_is.* -> xuap.* -> trajPoint.* The format of those files is explained below.

Then you tell the programm where the ptv_is.* files are and where the xuap.* and trajPoint.* files should be written.

FirstFile, LastFile, delta_t, viscosity should be clear.

When producing xuap.* files along trajectories moving cubic polynomials of 3d order are fitted to the raw particle positions. From this filtered positions, velocities, and accelerations are produced and written into xuap.* files. 21 is the possible maximum of points that can be used for such a fit. The minimum is 4. It will crash with less than 4 and more than 21.

Min left/right determines how much centered such a fit should be.

Max vel is hardly used, but acts as a safety measure. Typically it should be 3-5 times larger than r.m.s. u.

Radius of interpolation is a very important parameter. It determines how large the interpolation domain around a point x should be. As a rule of thumb the radius should not exceed 5 Kolmogorov units. Anything from 7-12 points per interpolation sphere is enough.

Depending on the weights the result aims to improve quality of relative error of $\text{diva} = -4Q$ and relative acceleration error of $Du/Dt = du/dt + u \cdot du/dx$.

When running the trajPoint part you should see something like the following (make sure that points per sphere are around 7-10, that the number of good points are around 40% or better, and that the values for r.m.s. u and dissipation are *plausible*).

```
C:\d:\technical\codes\post_process\release\post_process.exe
mean dissipation [m^2/s^3]...5.11815e-005

processing file .....10020
point per sphere.....8.44295
% rel. diva < 0.1.....40.3403
% rel. acc < 0.2.....43.6291
r.m.s. u [m/s].....0.0138089
mean dissipation [m^2/s^3]...5.03364e-005

processing file .....10021
point per sphere.....8.43017
% rel. diva < 0.1.....40.262
% rel. acc < 0.2.....43.8237
r.m.s. u [m/s].....0.013788
mean dissipation [m^2/s^3]...4.98872e-005

processing file .....10022
point per sphere.....8.41414
% rel. diva < 0.1.....40.2024
% rel. acc < 0.2.....43.7023
r.m.s. u [m/s].....0.0137654
mean dissipation [m^2/s^3]...4.99904e-005

processing file .....10023
```

Output format?

[Post-PTV software of Beat Lüthi](#)

The output is in two kinds of files. For each ptv_is.* there will be a xuap.* containing column by column the following (r=raw, f=filtered):

link_past, link_future, x_r,y_r,z_r,x_f,y_f,z_f,u_f,v_f,w_f,ax_f,ay_f,az_f,sucessfull

The main result is stored in the trajPoint.* files. Each trajectory that starts at frame x will be stored in trajPoint.x. This explains why the second trajPoint.* is so much larger than all the others. The format is best explained with a screen shot of where the output is written to file:

```
fprintf(fpp, "%lf\t", xp[ii]); //1
fprintf(fpp, "%lf\t", yp[ii]); //2
fprintf(fpp, "%lf\t", zp[ii]); //3
fprintf(fpp, "%lf\t", up[ii]); //4
fprintf(fpp, "%lf\t", vp[ii]); //5
fprintf(fpp, "%lf\t", wp[ii]); //6
fprintf(fpp, "%lf\t", axp[ii]); //7
fprintf(fpp, "%lf\t", ayp[ii]); //8
fprintf(fpp, "%lf\t", azp[ii]); //9
fprintf(fpp, "%lf\t", w1p[ii]); //10
fprintf(fpp, "%lf\t", w2p[ii]); //11
fprintf(fpp, "%lf\t", w3p[ii]); //12
fprintf(fpp, "%lf\t", s11p[ii]); //13
fprintf(fpp, "%lf\t", s12p[ii]); //14
fprintf(fpp, "%lf\t", s13p[ii]); //15
fprintf(fpp, "%lf\t", s22p[ii]); //16
fprintf(fpp, "%lf\t", s23p[ii]); //17
fprintf(fpp, "%lf\t", s33p[ii]); //18
fprintf(fpp, "%lf\t", utp[ii]); //19
fprintf(fpp, "%lf\t", vtp[ii]); //20
fprintf(fpp, "%lf\t", wtp[ii]); //21
fprintf(fpp, "%lf\t", daxdxp[ii]); //22
fprintf(fpp, "%lf\t", daxdyp[ii]); //23
fprintf(fpp, "%lf\t", daxdzp[ii]); //24
fprintf(fpp, "%lf\t", daydxp[ii]); //25
fprintf(fpp, "%lf\t", daydyp[ii]); //26
fprintf(fpp, "%lf\t", daydzp[ii]); //27
fprintf(fpp, "%lf\t", dazdxp[ii]); //28
fprintf(fpp, "%lf\t", dazdyp[ii]); //29
fprintf(fpp, "%lf\t", dazdzp[ii]); //30
fprintf(fpp, "%lf\t", quality); //31 0=good, 1=bad
fprintf(fpp, "%lf\n", (double)(ii)); //32 age along trajectory
```

More?

[Post-PTV software of Beat Lüthi](#)

The rest is up to you. We use such files to take it to Matlab or make HDF5 files from it. If you have question ask [Beat](#) , [Alex](#) or [Markus](#) .