Appendix

A.1 Handling of the 3D PTV software

Techniques of digital photogrammetry are suitable for static applications as well as for 3D measurement tasks in dynamic processes (El-Hakim, 1986). 3D PTV can be considered a typical application field of digital photogrammetric techniques (Grün, 1994). The method requires the acquisition and processing of stereoscopic, synchronized image sequences of the moving particles.

In the following sections a step by step advice for running PTV, some information on the input/output files is given.

A.2 Running the 3D PTV software under different operation systems

The source code of the PTV software is written in C, Tcl/Tk is required for the graphical user interface (GUI). The software can be compiled on any operation system where C and Tcl/Tk is available. The current PTV implementation runs under UNIX as well as Windows operating systems. For the image sequence data acquisition PCs with Windows operation system are used. The GUI is implemented with Tcl/Tk (current version is 8.4.2, released in march 2003, (Tcl, 2003)), which is freely available for Windows and UNIX platforms. Considering that most potential PTV users might only work with one of the mentioned operating system, the portability also could contribute to an easy distribution of the developed PTV method.

A.3 Parameter files for 3D PTV

Before running PTV, several parameter files (in ASCII) have to be created in a subdirectory of the project directory, it has to be called *parameter* and must contain the following files:

ptv.par	main parameter file
criteria.par	object volume and correspondences parameters
sequence.par	sequence parameters
targ_rec.par	parameters for particle detection
cal_ori.par	calibration plate, images, orientation files
detect_plate.par	parameters for control point detection
man_ori.par	point number for manual pre-orientation
orient.par	flags for camera parameter usage
track.par	tracking parameters
sortgrid.par	sortgrid parameter
pft_version	flag for peak fitting version

For storing of the results a directory named *res* should exist. Working with PTV the Tcl/Tk-GUI allows editing parameter files, to process a single time step or whole sequences. As a preparing steps a calibration of the image acquisition system is required. The main window is shown in Fig. 61. Each processing step can be started from the task bar, the processing status and the actual mouse button functions are displayed in the information window below. With

Start the project parameters are loaded from the input files and additional windows for the display of the images for each camera are opened.

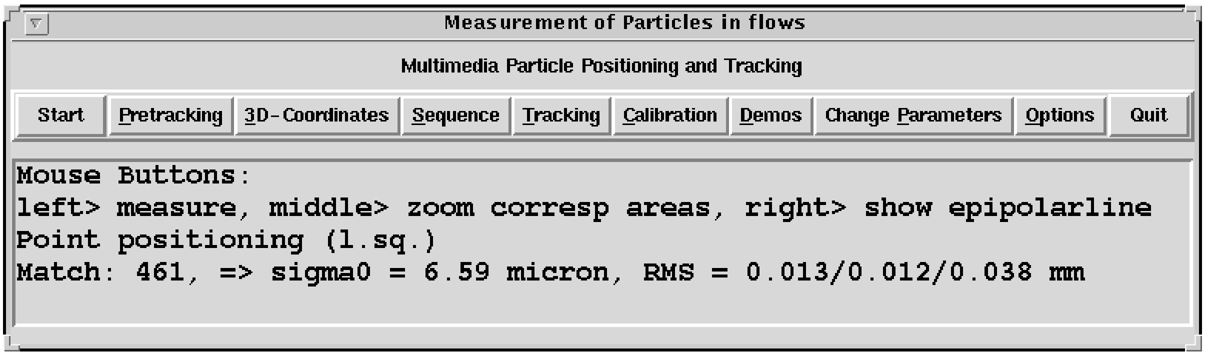

Fig. 61: Main window of PTV software

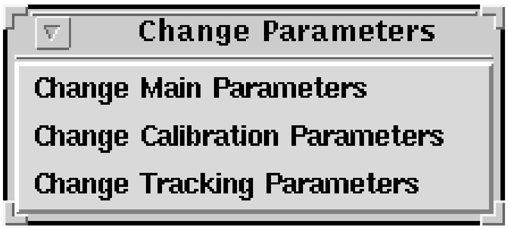

Fig. 62: Submenu Change Parameters

The settings for the PTV processing can be edited through three different parameter windows (to be opened under the submenu ***Change Parameters***, shown in Fig. 62). The windows for the editing of the parameters can be found under ***Change Main Parameters*** (shown in Fig. 63), ***Change Calibration Parameters*** (Fig. 64) and ***Change Tracking Parameters*** (Fig. 69).

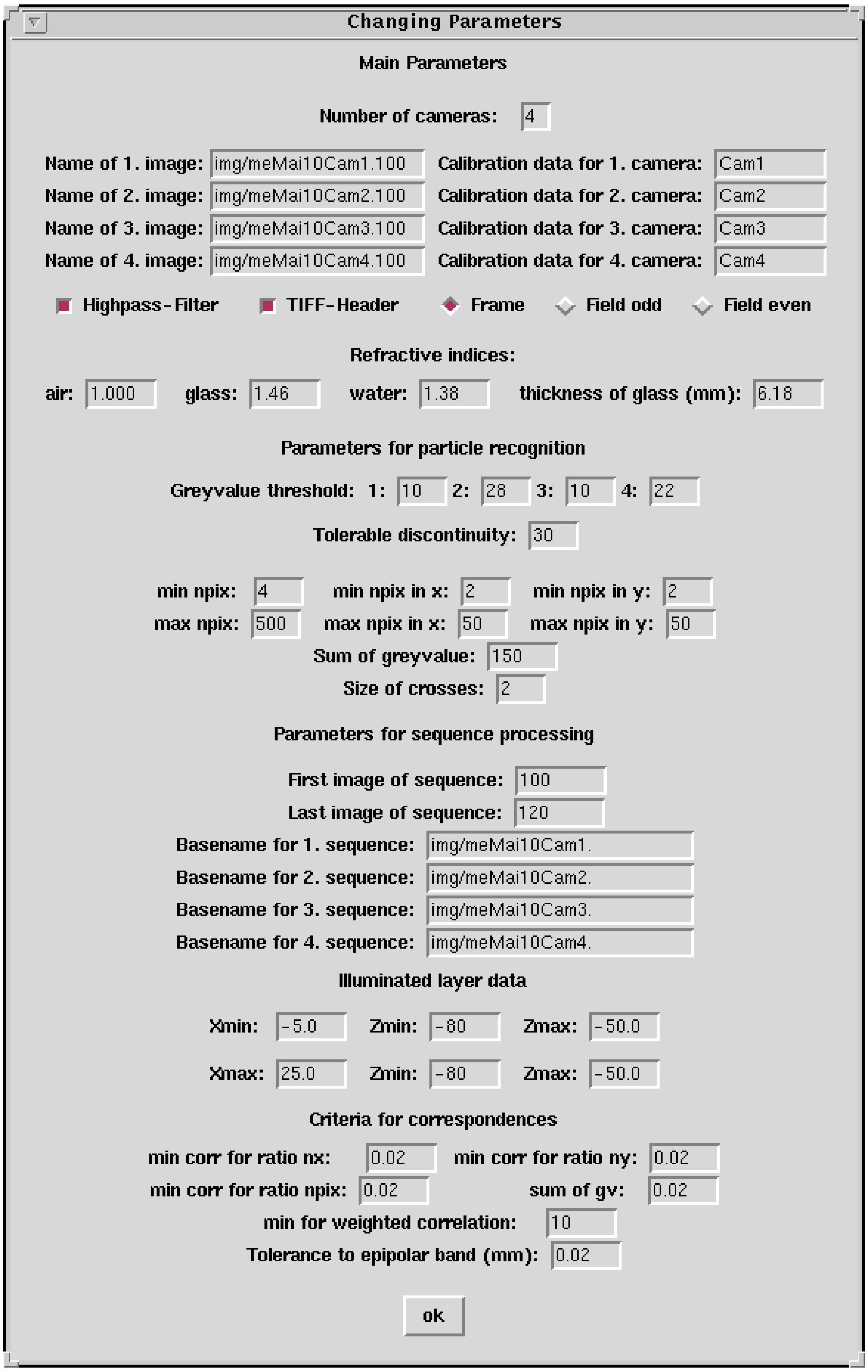

Fig. 63: Main parameter window

A.4 Data Input

The processing of image data with PTV requires for each camera a valid video sequence stored in TIFF- (8 Bits/Sample, single image plane) or as raw data files. The respective flag for TIFF or raw file must be set. The filename for the sequence image is a combination of a basename

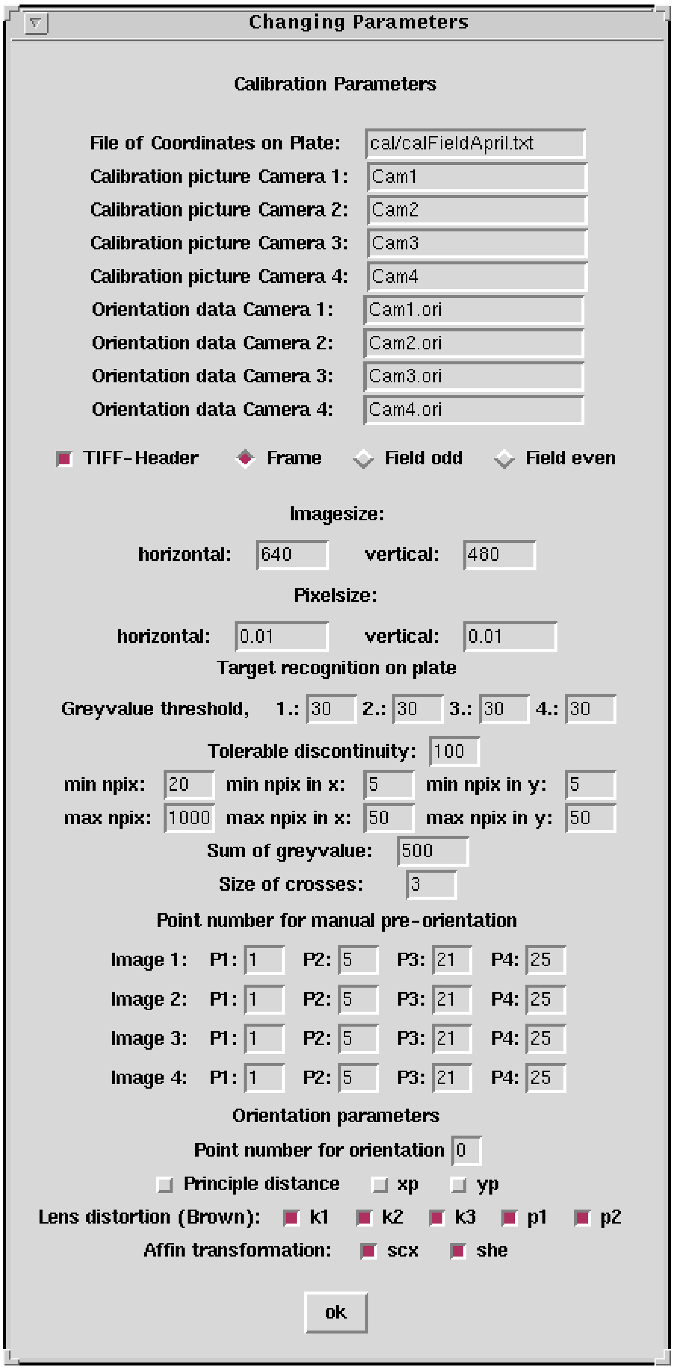

Fig. 64: Calibration parameter window

indicating the camera and a current number. For each camera an orientation data file (*name.ori*) and an additional parameters file (*name.addpar*) must exist and has to be accessible for PTV.

Examples for the parameterfiles:

<u>ptv.par:</u> main parameter file

```
4             number of cameras
cam3.100      image of first camera
kal1          calibration data of first camera
cam0.100      image of second camera
kal3          calibration data of second camera
```

cam1.100	image of third camera
kal4	calibration data of third camera
cam2.100	image of fourth camera
kal5	calibration data of fourth camera
1	flag for highpass filtering, use (1) or not use (0)
1	flag for TIFF header (1) or raw data (0)
720	image width in pixel
576	image height in pixel
0.009	pixel size horizontal [mm]
0.0084	pixel size vertical [mm]
0	flag for frame, odd or even fields
1.0	refractive index air [no unit]
1.5	refractive index glass [no unit]
1.0	refractive index water [no unit]
9.4	thickness of glass [mm]

===

criteria.par:	object volume and correspondence parameters
0.0	illuminated layer data, xmin [mm]
-10.0	illuminated layer data, zmin [mm]
0.0	illuminated layer data, zmax [mm]
10.0	illuminated layer data, xmax [mm]
-10.0	illuminated layer data, zmin [mm]
0.0	illuminated layer data, zmax [mm]
0.02	min corr for ratio nx
0.02	min corr for ratio ny
0.02	min corr for ratio npix
0.02	sum of gv
33	min for weighted correlation
0.02	tolerance to epipolar line [mm]

===

sequence.par:	sequence parameters
cam0.	basename for 1. sequence
cam1.	basename for 2. sequence
cam2.	basename for 3. sequence
cam3.	basename for 4. sequence
100	first image of sequence
119	last image of sequence

===

targ_rec.par:	parameters for particle detection
12	grey value threshold 1. image
12	grey value threshold 2. image
12	grey value threshold 3. image
12	grey value threshold 4. image
50	tolerable discontinuity in grey values
25	min npix, area covered by particle
400	max npix, area covered by particle
5	min npix in x, dimension in pixel
20	max npix in x, dimension in pixel
5	min npix in y, dimension in pixel
20	max npix in y, dimension in pixel
100	sum of grey value
1	size of crosses

===

<u>cal_ori.par:</u> calibration plate, images, orientation files

ptv/ssc_cal.c3d	control point file (point number, X, Y, Z in [mm], ASCII
kal1	calibration image camera 1
kal1.ori	orientation data camera 1
kal3	calibration image camera 2
kal3.ori	orientation data camera 2
kal4	calibration image camera 3
kal4.ori	orientation data camera 3
kal5	calibration image camera 4
kal5.ori	orientation data camera 4
1	flag for TIFF header (1) or raw data (0)
0	flag for frame (0), odd (1) or even fields (2)

==

<u>detect_plate.par:</u> parameters for control point detection

30	grey value threshold 1. calibration image
30	grey value threshold 2. calibration image
30	grey value threshold 3. calibration image
30	grey value threshold 4. calibration image
40	tolerable discontinuity in grey values
25	min npix, area covered by particle
400	max npix, area covered by particle
5	min npix in x, dimension in pixel
20	max npix in x, dimension in pixel
5	min npix in y, dimension in pixel
20	max npix in y, dimension in pixel
100	sum of grey value
3	size of crosses

==

<u>man_ori.par:</u> point number for manual pre-orientation

28	image 1 p1 on target plate (reference body)
48	image 1 p2
42	image 1 p3
22	image 1 p4
28	image 2 p1
48	image 2 p2
42	image 2 p3
23	image 2 p4
28	image 3 p1
48	image 3 p2
42	image 3 p3
22	image 3 p4
28	image 4 p1
48	image 4 p2
42	image 4 p3
22	image 4 p4

==

<u>orient.par:</u> flags for camera parameter usage 1=use, 0=unused

2	point number for orientation, in this case every second point on the reference body is used, 0 for using all points
1	principle distance
1	xp

```
1                       yp
1                       k1
1                       k2
1                       k3
0                       p1
0                       p2
1                       scx
1                       she
=========================================================================
```

<u>track.par:</u> tracking parameters

```
-0.8                    dvxmin
0.8                     dvxmax
-0.8                    dvymin
0.8                     dvymax
-0.8                    dvzmin
0.8                     dvzmax
120                     dangle
0.4                     dacc
1                       add, flag for additional particles use (1)
=========================================================================
```

<u>sortgrid.par:</u> changes must be done with an editor

```
5.0                     distance between detected image coordinate
                        and reprojected control point in pixel
=========================================================================
```

<u>pft_version:</u>

```
3                       flag for peak fitting version (newest)
=========================================================================
```

Those parameter files can be edited under ***Changing Parameters*** in the appropriate submenu windows (***Main Parameters*** shown in Fig. 63, ***Calibration Parameters*** shown in Fig. 64 , ***Tracking Parameters*** shown in Fig. 69).

A.5 Examples for *name.ori* and *name.addpar* files

<u>name.ori:</u> camera orientation file

```
101.8263 -9.9019 65.1747                projective center X,Y,Z, [mm]
0.4151383 -0.0069793 1.5073263          omega, phi, kappa [rad]

0.0634259 -0.9979621 -0.0069792         rotation matrix (3x3)
0.9130395  0.0608491 -0.4033067         [no unit]
0.4029095  0.0192078  0.9150383

-0.6139  -0.0622                        xp, yp [mm]
8.7308                                  principle distance [mm]
```

<u>name.addpar:</u> additional parameters

```
-0.00342 0.00007 -0.00002 -0.000545 0.000632 1.004600 0.002856
```

```
k1 [no unit], k2 [no unit], k3 [no unit], p1 [no unit], p2 [no unit], scale
in x [no unit], shearing in [rad]
```

A.6 Calibration of the image acquisition system

Before an image sequence can be processed the calibration of the image acquisition system has to be performed. As a result of the calibration the three coordinates X_O, Y_O, Z_O of the projective center and three angles ω, φ, κ describing the direction of the optical axis are determined (exterior orientation). To meet the physical realities the following parameters and model extensions are introduced (interior orientation and additional parameter):

- The camera constant c (computational imaging width).
- The principle point coordinates x_H, y_H.
- The lens distortion is modelled with up to five parameters (k_1, k_2, k_3, p_1, p_2).

Different clock rates of camera and framegrabber cause a horizontal scale factor s_x, which deviates from one. A further parameter δ is used for the shearing effect.

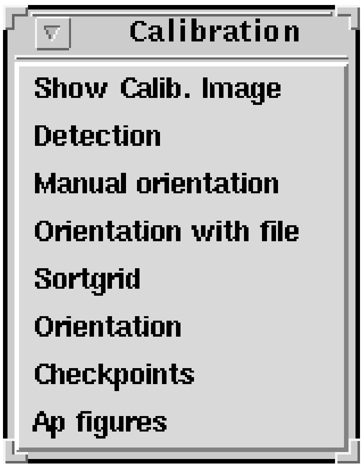

Fig. 65: Submenu with the steps of calibration procedure

In total 16 parameters (exterior orientation, interior orientation, additional parameters) can be used to model each camera. In the submenu ***Calibration Parameters*** (Fig. 64) the referring flag for each parameter can be set. If the parameter flag is set the referring value is determined in the bundle adjustment. Otherwise it is introduced as a fixed parameter. The ***Calibration*** submenu (Fig. 65) contains the following steps:

A suitable set of calibration images must exist as TIFF (8 Bits/Sample, single image plane) or as raw data files. The respective flag for TIFF- or raw file must be set.

To load the images for the calibration press ***Show Calib. Image***. For the acquisition of the calibration images use a reference body suitable for the object volume.

The determination of the image coordinates of the reference points is started with ***Detection***. It is the application of the peakfitting routine after a high pass filtering. The relevant parameters for the detection have to be set in the ***Change Calibration Parameters*** window (Fig. 64).

With ***Manual Orientation*** four points of the reference body can be measured manually in each calibration image. A file named *man_ori.dat* with the measured image coordinates is created. The points that have to be measured are displayed in the information window of the main menu (Fig. 61). The measurements are required for each calibration image, once measured the image coordinate are stored in the *man_ori.dat* file.

If the file *man_ori.dat* already exists the measurement could be reused with ***Orientation with file***.

The automated points detection might produce mismatches e.g. due to scratches and other disturbances one the reference body. Points that will lead to mismatching in the ***Sortgrid*** routine should be deleted in advance.

With ***Sortgrid*** all points of the reference body are reprojected in image space using the approximate camera orientation data. The search radius for the detected points is defined in *parameters/sortgrid.par*. The value of the search radius in pixels has to be changed in the file itself, as it cannot be set over the GUI of the PTV software

Orientation calculates the parameters of the cameras in a bundle adjustment. As the approximation values for the orientation parameters might be unsuitable their determination probably have to be performed stepwise. The user should also avoid overparametrization as not all unknowns might be determined reliably.

Checkpoints shows residual in the control points. The display of the residuals can be used for a visual control of the orientation results.

Ap figures shows the effect of the additional parameters. The effect of the additional parameters is enlarged by a certain factor for visualization purpose.

A.7 Processing of a single time step

Fig. 66: Submenu Pretracking

Under ***Pretracking*** (Fig. 66) the processing of a single time step regularly starts with the application of a highpass filtering (***Highpass***).

After that the particles are detected (***Image Coord***) and the position of each particle is determined with a weighted grey value operator.

The next step is to establish correspondences between the detected particles from one camera to all other cameras (***Correspondences***).

With pressing of ***3D-Coordinate*** on the task bar of the main window the determination of the three-dimensional positions of the particles in object space is performed. The 3D data is written to disc, if existing into the directory named *res,* otherwise into the current directory. The processing of a single time step is necessary to adjust parameters like grey value thresholds or tolerance to the epipolar line.

A.8 Display of image sequences

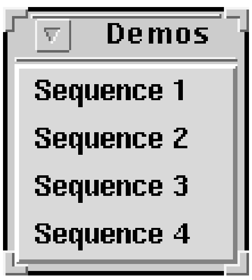

Fig. 67: Display of image sequences

Under the submenu ***Demos*** (Fig. 67) the display of the image sequence recorded by each cameras can be started. After the choice of the camera number the sequence is displayed in the referring image display window.

A.9 Processing of image sequences

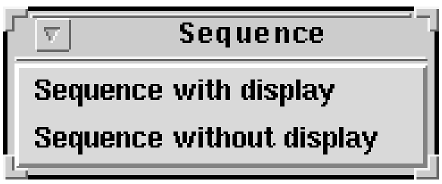

Fig. 68: Sequence processing

After having optimized the parameters for a single time step the processing of the whole image sequence can be performed under ***Sequence*** (Fig. 68) with or without display of the currently processed image data. It is not advisable to use the display option when long image sequences are processed. The display of detected particle positions and the established links can be very time consuming.

For each time step the detected image coordinates and the 3D coordinates are written to files, which are later used as input data for the ***Tracking*** procedure.

A.10 Tracking of particles

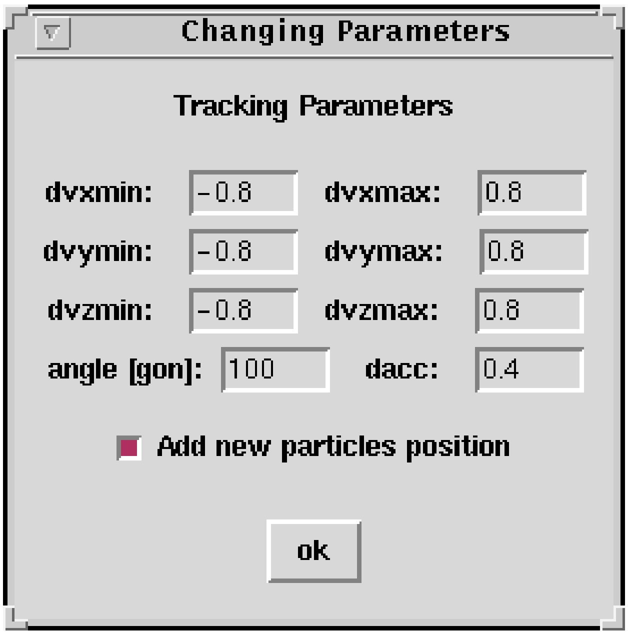

Fig. 69: Tracking parameter window

Before the tracking can be performed several parameters defining the velocity, acceleration and direction divergence of the particles have to be set in the submenu ***Tracking Parameters*** (Fig. 69). The flag '*Add new particles position*' is essential to benefit from the capabilities of the enhanced method.

To derive a velocity field from the observed flow ***Tracking with display, Tracking without display*** and ***Tracking backwards*** under ***Tracking*** (Fig. 70) has to be performed. Again it is not advisable to use the display option if long sequences are processed. The tracking procedure allows bidirectional tracking.

Detected Particles displays the detected particles from the sequence processing.

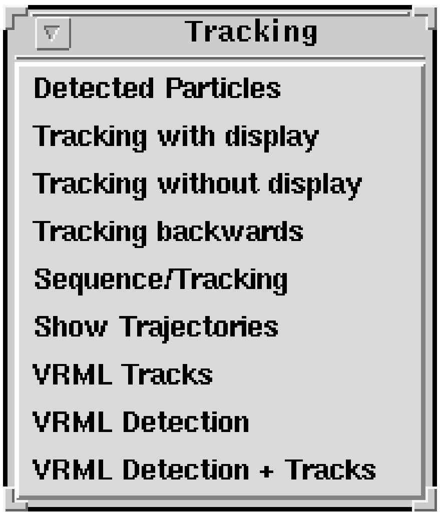

Fig. 70: Submenu Tracking

For a combined sequence processing and tracking in forward direction ***Sequence/Tracking*** can be started, ***Tracking backwards*** is not performed in this procedure.

Show Trajectories displays the reconstructed trajectories in all image display windows.

For the generation of VRML-files of the results from the sequence processing press ***VRML Detections*** to get an untracked point cloud, ***VRML Tracks*** to get exclusively the trajectories or ***VRML Detection + Tracks*** to get all 3D points and links. The VRML output is created in VRML version 1.0.

A.11 Options

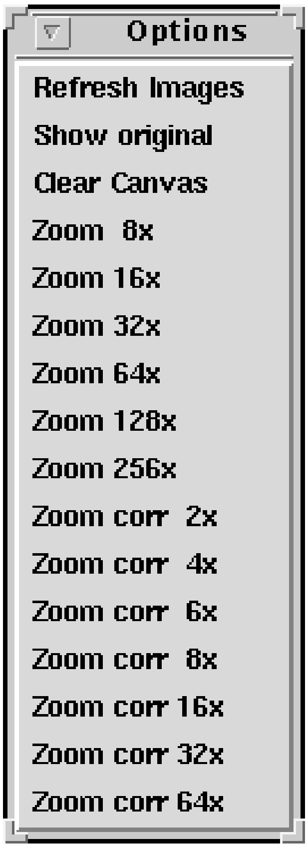

Fig. 71: Image display and zoom options

Several Options for image display and zooming are accessible under ***Options*** (Fig. 71). Zooming is either performed for each image individually or in corresponding image regions.

A.12 Visualization of tracking results

For a visual analysis of the results three different visualization tools were used. VRML-output can be viewed in a common VRML-Browser, the output can be directly generated by the PTV implementation allows a 3D display of the trajectories in a static way.

A VRML-representation was also used for the visualization of single trajectories or velocity field where the particle types (epipolar intersection method, added particles) should be distinguished.

An alternative visualization, which also allows showing the motion of the particles was realized in MATLAB. The input data for this purpose cannot directly be generated by the PTV implementation, but by a small format conversion program. The MATLAB visualization tool was used to show the complete velocity field or selected trajectories of the regarding data set.

The most suitable velocity field visualization is possible by a software tool developed by Nicola D’Apuzzo (D’Apuzzo, to appear). With a GTK-GUI is can show the velocity field in static and dynamic way in different modes. During the display of the particle motion the viewpoint can be changed interactively. Single trajectories or region of the velocity field can be selected and visually analysed in detail. This tool is very helpful for a qualitative assessment of the results of the PTV processing. Remarkable is its performance considering the visualization of large data sets. Fig. 72 shows the GUI of the software tool to visualize the tracking results.

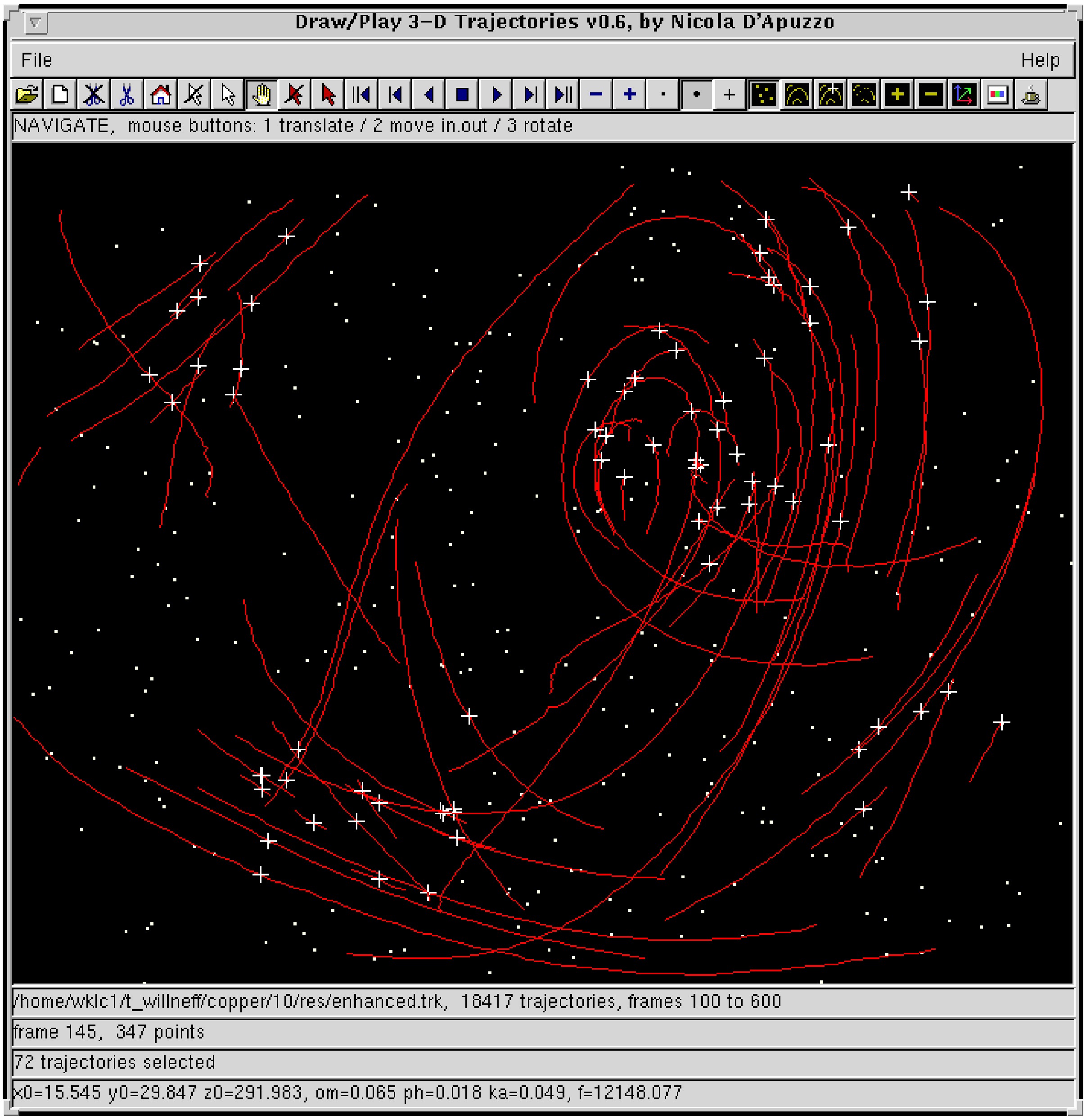

Fig. 72: Software tool for the visualization of tracking results, developed by D'Apuzzo