DigImage

Image Processing for Fluid Dynamics

Two-dimensional Particle Tracking

Stuart Dalziel

DL Research Partners

1993 – 2001

Contents

1 Concept

A wide range of techniques have been developed over the years for obtaining qualitative and quantitative measurements of fluid flows. Traditional visualisations are primarily qualitative, yielding the structures present within some finite two or three dimensional domain within the flow. Any quantisation required laborious measurement by hand, typically from photographic film. Hot wire and hot film anemometry (HWA) and laser Doppler anemometry (LDA) yield precise velocity measurements at one or more isolated points within the flow. The inherent Eulerian nature of such measurements does not allow ready access to the Lagrangian character of the flow, nor do they offer much insight into the structures present.

A number of techniques have been developed to overcome the limitations for HWA and LDA. Scanning and multiple probes allows HWA and LDA simultaneous (or nearly so) measurements at a large number of points within a flow. Unfortunately physical limitations mean the number of points remains relatively small and the cost of the equipment required high. An alternative is to base the measurement technique on a multi-dimensional visualisation of the flow. The most successful class of methods to date fall under the heading of particle image velocimetry (PIV). In PIV, the flow is seeded with small, (nearly) neutrally buoyant particles which are assumed to follow fluid elements without affecting the flow itself. The flow is illuminated in some manner such that all the particles in some finite domain are visible. A series of images of the flow are captured on some medium (*e.g.* film or video tape) to record how the particles move in response to the fluid flow. These images are then analysed in order to determine how the particles, and hence the fluid elements, move in time.

There are two main techniques which may be employed to gain information from successive images containing individually identifiable particles, where the particles may have moved a significant distance (compared with the size of the particles): image cross-correlation (pattern matching PIV) and particle tracking. In the first of these an image of a two-dimensional flow (or a two-dimensional slice of a three-dimensional flow) at time t_n is divided into a large number of possibly overlapping cells. The correlation function between each cell at $t = t_n$ and a similar cell at $t = t_{n+1}$, subject to some translation in the plane of the image, is evaluated. This evaluation may be based on either the complete image within the cell or the location of discrete particles. The velocity of a cell is evaluated from the cell translation which optimises the correlation between the two images. In the simplest case, the optimal translation will be that which maximises the correlation function. Prior to this correlation procedure, the image may be *enhanced* in some manner to remove the effects of background noise. The enhancement procedure may be simply locating the individual particles.

The strengths of the cross-correlation method are that is fairly robust to noise and has excellent velocity resolution (the accuracy with which displacements may be obtained is a function of the cell size and the distribution of features within it rather than the pixel resolution). The spatial resolution is inversely proportional to the cell size: the overall data quality is thus a compromise between velocity and spatial resolution. The main disadvantages are the considerable time required to compute the optimal correlation and the inability to cope with any structure across the illuminated plane (*i.e.* velocity gradients parallel to the viewing direction). In general the method does not allow individual particles to be tracked, and hence has no immediate access to Lagrangian descriptions. However, it is a relatively simple matter to add some degree of particle tracing once the velocity field is known, and hence access the Lagrangian nature of the flow.

Particle tracking offers a more fundamental approach to PIV. There are two main approaches which are exactly equivalent to the manual methods of analysing streak (or multiple exposure) photographs and multiple (time series) photographs. In the streak photograph method, the effective camera shutter is opened for a long time during which the particles move many particle diameters. This long exposure may be produced directly with a suitably slow shutter speed, or synthesised by combining multiple exposures (*e.g.* ORing a sequence of video frames using a digital frame grabber with a shutter speed equal to the field rate - the DigImage option `[;UP Particle streaks - bright particles]` and macro `STREAK.CMD` both perform such an operation). Once the streaks have been produced, image processing techniques may be applied to locate them and analyse their shape, orientation etc.

The alternative of utilising a time series of images offers a greater volume of information on the particle positions as a function of time, especially in the context of digital image processing where quantisation yields a relatively low spatial and intensity resolution. Knowing the approximate location of a particle at a relatively large number of times enables a much more accurate estimation of the position of a particle at a given time, and of its velocity, provided the sampling frequency is much higher than the highest frequency in the particle motion. To make use of this information some method must be developed for tracking particles from one image to the next. In the limit of particles moving only a small fraction of their diameter between each sample, the process of matching particles in one image with their position in the next image is straight forward - the particle images closest together in two adjacent samples will correspond to the same physical particle. However, if the particles may move many diameters between samples, more sophisticated algorithms must be employed.

The algorithm used in the matching process may utilise spatial and temporal information in addition to particle characteristics and prior knowledge of the flow. Generally, only some of these features will be needed to determine which particle image is which particle. For example, if spatial correlation is not utilised, then two-dimensional projections of three-dimensional flows with significant velocity gradients parallel to the direction of viewing, may be analysed (recall that cross-correlation techniques are unable to cope with such images). Moreover, the basic approach is not limited to a two-dimensional projection of a three-dimensional flow but is capable of full three-dimensional analysis. By applying the matching process repeatedly, time-series for individual particles may be obtained to describe some of the Lagrangian nature of the flow.

The accuracy with which the velocities may be measured is limited by the accuracy with which the individual particle images may be located and the time period over which the velocity may reasonably be evaluated (this must be shorter than the period corresponding to the maximum frequency in which you are interested). The accuracy of location depends in turn on the particle size and the method used to determine their positions. In general, the velocity resolution will be less than that for the cross-correlation approach, but is nevertheless adequate in most situations. The spatial resolution is limited primarily by the number of particles in the flow: the more particles, the higher the resolution. In practice the resolution of video technology and the frame grabber imposes the most stringent limitation on the number of particles able to be tracked. Eulerian as well as Lagrangian descriptions may be obtained, utilising a suitable interpolation method, if the particle seeding density is sufficiently high.

This document outlines and describes the two-dimensional particle tracking technique utilised by DigImage. This method represents an efficient, reliable approach to tracking particles from a two-dimensional projection of a flow. The computation required to analyse each frame pair increases only slightly faster than linearly with the number of particles, allowing between 5 and 30 frame pairs per minute to be analysed. A step by step tutorial is also provided in the file `DigImage\Macros\Track.CMD`. This macro may be used either with the sample tracking movie available from `http://tiki.damtp.cam.ac.uk/digimage/examples/tracking/`, or by printing it out as step by step instructions on how to set up Dig*Image* for particle tracking.

2 Dig*Image*: Particle Tracking

Particle tracking under Dig*Image* may be broken down into three separate phases: experimentation, tracking and subsequent analysis. We shall discuss first the second phase, the particle tracking, as the experimental method and subsequent analysis depend heavily on how the information is gathered.

There are three main processes required to track particles: the images of the flow must be captured, the particles must be located within these images, and the relationship between particles in successive images must be determined.

2.1 IMAGE CAPTURE

Dig*Image* uses Super VHS video tape to record a single view of an illuminated region of an experiment. During the tracking phase, the video tape is replayed and the images are captured by digitising the video signal. As only four or sixteen (depending on the hardware) complete frames may be captured in one go, it is necessary to be able to control the video recorder and synchronise the frame grabber with it so that exactly the right series of four (or sixteen) frames may be acquired in one go. The video recorder must be in PLAY mode during the frame acquisition as small errors in the synchronisation when in PAUSE mode lead to unacceptable images; single frame stepping through the tape is therefore not feasible.

The task of controlling the video recorder so that exactly the right frame is acquired is not trivial. Even VTRs designed for precise editing applications occasionally lose track of a frame when changing direction. As an experiment may be tracked 5000 or more frames (the software limit is 65279 samples or the length of the video tape), the tape may need to change direction well over 1000 times, so even small timing errors will accumulate. Moreover, as the velocities are determined by what is effectively a finite difference in time, any errors will be reflected in the measured velocity.

Dig*Image* makes use of a Panasonic AG-7330, Panasonic AG-7350 or JVC BRS822 Super VHS video tape recorder. These three machines are fitted with an RS-232 interface allowing computer control of all the video functions. The interface also allows the computer to interrogate the VTR to determine what it is doing and where it is. Unfortunately, the VTR is able to return its position only in an asynchronous manner which may lag behind the true position of the VTR. Moreover with the AG-7330 the position is limited to units of one second, and with both Panasonic machines there is the possibility of the VTR returning an incorrect time. As we require to know the exact frame and the precise moment to acquire the buffers this RS232 control is not adequate. The RS-232 interface for the Panasonic machines has therefore been modified to make the VTRs internal video field strobe available on a spare pin (see installation documentation). On the AG-7330 the field strobe undergoes a transition every video field (1/50 or 1/60 of a second), while on the AG-7350 it pulses once a frame (1/25 or 1/30s). An interrupt driven routine in the host computer counts these transitions, so that, when provided with the direction of tape transport, the tape position may be determined. For the JVC a complicated procedure utilising a frame grabber generated interrupt is employed. During critical operations, these interrupt driven routine must have a higher priority than any other process in the host PC, otherwise it is possible to miss a transition. Interrupts in both the real and protected modes for the CPU are employed.

This control system would be all that was required if the field strobe in the VTR always changed state when it should. However this is not the case with the cheaper Panasonic machines. The AG-7330 always loses two fields when changing from reverse to forwards (an error which can be catered for), and both machines occasionally (about one time in ten direction changes) loses a further two pulses. It is these intermittent errors which cause the difficulties. To overcome this, before processing a video tape, DigImage pre-formats the tape with a time-code pulse on one of the audio tracks (normally channel 1). A short tone pulse is recorded every eight video fields (four frames). During subsequent tape operations, the relative position of these pulses is used as a check for the field count. If the position of the pulses appears to have changed since the previous tape operation, DigImage is alerted to an error in the field counter; this error is then corrected before any image acquisition or precision tape positioning.

Extensive tests of this control system have shown DigImage to be 100% reliable in capturing the required frame, *provided* the tape is of adequate quality and the audio pulses have been recorded correctly. The system operates through a customised serial cable linking the modified RS-232 interface, audio out and audio in connectors on the VTR with the `COM1:` and `COM2:` (or `COM3:` and `COM4:`) ports on the PC. Details of these components are given in the Installation Guide. No special expansion cards are required for the PC.

An alternative approach to controlling the video source would be to utilise a laser disk to store the images. There are two draw backs of this option: the cost and the resolution. The current generation of laser discs sufficiently sophisticated for these applications cost in the region of £13000-00. The cost of replacement write-once discs is around £250-00 for 32 minutes. This compares very poorly with around £1600-00 or the AG-7350 and interface. Laser disks record the video signal in an analogue form using the PAL or NTSC standard for encoding colour information. While the recording medium has a high noise immunity, it does not offer any significant improvement in resolution over Super VHS.

A second alternative for slower flows is to record the video sequence directly to the computer's hard disk using the `[;KM: Movies]` facility of DigImage. By avoiding recording the signal on an alalogue medium, the signal to noise ratio of such direct to disk recordings will be better, and as a consequence the quality of the results will be improved. The main limitation with this approach is that the sampling rate is limited and dependent on the size of the window being recorded. For more information, consult the System Overview .

Before processing the images in any way, the permanent reference points set up by `[;PR: Reference points]` (refer to the System Overview for details) are located within each image. If the root mean square error in the mapping which is generated exceeds the limit set by `[;USZ Limit on rms error for mapping]`, then the image will be rejected as being of inadequate quality. If the captured image for this frame has not been rejected too often before (how often depends on a second setting in `[;USZ Limit on rms error for mapping]`), then DigImage will try to improve the quality of the image by repeating the acquisition process until it has either captured a satisfactory image, or the limiting number of retries has been reached.

After an image has been captured and reference points validated, it may be necessary to enhance the quality of the image in some manner. Typically this requires some form of filtering. A wide range of low and high pass filters are available under DigImage (see help facility in `[;USI: Filter type]` for more details), though it is seldom desirable to use these. Rather it is far better to ensure clean experimental images than to throw away information by trying to remove unwanted noise. One occasion where filtering may be desirable is in the removal of a dynamically varying background. Static background variations may be removed by `[;USB: Background removal]`. While most of the options in `[;USI: Filter type]` take the form of convolution filters, `[;USIU User supplied pre-process subroutine]` allows arbitrary pre-processing to be performed using a user-supplied subroutine which hooks onto the particle tracking. The default routine simply removes large scale variations in the background image, effectively implementing a more sophisticated dynamic equivalent of `[;USBR Remove particles then low pass filter]` to construct a background image which is then subtracted from the raw image. Complete details on the use of this option and other customisable procedures are given in the file `Document\Trk2Hook.DOC`.

One of the hidden problems with using video tape is that typical CCD cameras have electronic shutters which are open on a cycle of 50Hz (or 60Hz), whereas a complete video frame is produced at only 25Hz (30Hz). Thus the information on one half of the interlace (the even lines) corresponds to an earlier time than the information on the other half of the interlace (odd lines). For flows which are evolving slowly, this need not be of concern as the particles will have moved only a small fraction of their size between the two halves of the interlace. Their position may be considered as the mean of the even and odd line positions.

In contrast, if the flow velocities are large so that a particle moves one or more particle diameters between the two halves of the interlace, then the particle will appear to be in two positions at once. Under these circumstances it may be necessary to reduce the vertical resolution from 512 lines to 256 lines, utilising only one half of the interlace to determine the particle positions.

If a high frequency response is required or high velocities are present, then the two halves of the interlace may be considered as separate snap shots of the flow, allowing sampling at 50Hz (60Hz for NTSC) to resolve 25Hz (30Hz) signals. To achieve this frequency response for the vertical component of velocity, the amplitude of the fluctuations must be much greater than the pixel size, otherwise the one line vertical offset between the even and odd lines will contaminate the results.

Some forms of video camera are able to produce an interlaced signal for which both video fields contain information at the same time. The Cohu 4910 series and Sony XC-77RR series, when combined with a phase-locked mechanical shutter are able to produce full resolution images. These cameras provide a much greater range of facilities than most conventional video cameras, yet cost little more. We therefore strongly recommend one of these cameras for particle tracking - further details are available in the file `Document\Cameras.DOC`.

In many cases, even with very clean experiments, there remains some residual background illumination. This may be either intentional or unintentional. Unintentional spatially uniform illumination presents no difficulty, but variations in the background across the viewed region need to be catered for. DigImage is able to allow for static or dynamic variations in the background illumination, and optionally gather additional information from the dynamic variations.

The basic technique for static background variations is to subtract some idealisation of the background from the incoming video signal. In the simplest situation, the idealisation may be a uniform intensity field. A number of options for determining the background illumination are presently available:

Feature access	Description
[;USBA ALU operations with buffer]	Under this option more general ALU operations may be used to modify an in-coming image. This option could be used to achieve the same effect as [;USBM Mask region with buffer] by selecting a ALU AND function, however any ALU function may be used in combination with a user-specified image.
[;USBN No background removal]	This assumes the background is uniform black or white.
[;USBF Low pass filter average picture]	This option captures and averages a number of images at some specified time during the experiment. The average is then passed through a simple low-pass filter.
[;USBM Mask region with buffer]	This option is designed specifically to enable tracking in nonrectangular domains. The effect is similar to using [;USBA ALU operations with buffer] with a binary image in a AND operation. Regions in the image to be processed corresponding to regions in the mask buffer with an intensity of 255 will be retained, while those regions corresponding to a zero in the mask buffer will be discarded. In this way the desired area may be selected from a large window in a manner more flexible than the basic rectangular windowing. Note that the data may alternatively be windowed or masked from within Trk2DVel if the necessary.
[;USBP Polynomial fit to background]	Under this option the average picture is broken into 10x20 pixel blocks with the average intensity being computed for each block. A bi-quartic polynomial is fitted (using a least squares routine) to these blocks. This least squares fit is then used as the background.
[;USBR Remove particles then low pass filter]:	This represents a more sophisticated version of the low pass filter. The average picture is first copied. The copy passes through a low-pass filter and is then subtracted from the original to produce a residual picture. The residual picture will give some indication of points, such as particles, in the average picture which need to be removed before creating the background. A constant is subtracted from the residual picture before it, in turn, is subtracted from the original averaged picture. Finally, this corrected average is passed through a low pass filter.
[;USBU User supplied buffer background]	With this final option, the user may specify a "background" image to be subtracted from the incoming image. This option is effectively a subset of [;USBA ALU operations with buffer].

Dynamic background removal is a form of high pass filtering and is covered by the [;USI: Filter type] menu with user-definable routines discussed in Document\Trk2Hook.DOC.

Temporal changes in the background may be used to either discriminate different regions of the flow, or (provided no more than 511 particles are to be tracked and the [;UG Start particle tracking - 511 particles]) to gather information about some scalar field (*e.g.* density marked with a fluorescent dye). Briefly the options are as follows:

Feature access	Description
`[;USRB Record background intensity]`	Available only if 511 or fewer particles to be tracked. Under this option the local background intensity in the neighbourhood of each particle is evaluated and assigned to one of eight possible levels between two user-defined limits. This information is stored along with the other data for each and every particle. The division into only eight levels is both to minimise the additional storage requirements for this information, and in the realisation that a greater degree of precision in determining the true local background intensity is unlikely. The parameters controlling this process are set up by the `[;USLB: Background intensity evaluation]` menu.

`[;USRE: Exclude region inside contour]`
 	
`[;USRI: Include region inside contour]`
	Rather than recording information about the background intensity field, this option provides a method of distinguishing two separate regions of the flow and performing the tracking process in only one such region. Typically the two regions will be marked with some visible tracer such as a fluorescent dye. At regular intervals **Dig*Image*** will locate an intensity contour within the image after utilising a special range of filters to remove the influence of particles on the image. The contours may optionally be filtered to produce a smooth boundary between the two regions (each region may manifest itself as more than one object in a given image). Finally, which is applied to the parent image to select only the desired regions pick up particles in the regions outside the contour.

`[;USRN No region recording]`	Of course, there is no need to utilise or record any information about variations in the background intensity if it is of no interest in a given flow.
`[;USRR: Record contour region particle in]`	Available only if 511 or fewer particles are being tracked. This option utilises the same mask image as is produced to Include/Exclude regions inside a contour, but rather than masking off certain regions prior to tracking, the mask image is used instead to flag which region the particle was in at a given time. The information thus produced is therefore similar to that obtained by recording the background intensity, but utilising only a binary approximation to the background intensity. This option is intensity dividing two distinct regions, while the previously mentioned option is more suited to relatively slow, continuous variations in the background.

Once the image has been corrected for static variations in the background intensity, and any masks applied, it is necessary to locate the particles within the image.

2.2 PARTICLE LOCATION

A wide variety of techniques exist for locating features within an image. For **Dig*Image*** the philosophy has been to *keep it simple*, partly as sophisticated location strategies do not produce significantly superior results, and partly on the grounds of computational efficiency. It is generally better to start with high quality images of an experiment, than to try to extract data from low quality images by throwing away some of the information.

A popular method of determining where a particle is to fit a Gaussian intensity distribution to the particle image and then determine the centroid of the Gaussian. The philosophical draw back of this approach is that real particles are unlikely to produce Gaussian intensity profiles, particularly as they are probably illuminated from a direction in the plane normal to the viewing direction and hence are illuminated only from one side. Moreover, the image of a particle will generally be relatively small, only a few pixels in size. The amount of information present does not therefore warrant the fitting of a Gaussian profile, especially as some of the information will need to be thrown away.

Any particle location routine should use as much of the available information as possible. Provided the same information and method is used at each successive time for a given particle, the change in the particle position will be represented as accurately as possible, even if the absolute particle position is not quite exact. Dig*Image* employs two methods of locating particles, one which uses all the information, and the other which summarises some of the information. The location of a particle is determined using both these methods, the results being compared to ensure consistency.

A particle in Dig*Image* is defined as an area of an enhanced image (see previous section) satisfying a number of criteria, based on the intensity, size and shape of the particles. The most basic criterion is the intensity which is used to identify potential particles or *blobs* within an image. For the purposes of this discussion, we shall assume that we are tracking light particles on a dark background. For dark particles on a light background, Dig*Image* inverts the incoming video signal to ensure this is always true.

Blobs are first located by searching for regions within the tracking window of an enhanced image which satisfy a spatially uniform intensity threshold (specified by `[;USLU Upper threshold]` or determined automatically - see below). As the "background" illumination may have been subtracted, this operation is formally equivalent to searching for regions which have an intensity greater than or equal to the background value plus the threshold. When a region satisfying the threshold criterion has been found, it is marked (as found) and its properties determined to see if it is in fact a particle. Dig*Image* makes a number of passes when searching for blobs. The number of passes is specified by `[;USLV Number of threshold levels]`, each with a lower threshold until the lower threshold specified by `[;USLT Lower threshold]` is reached. Some classes of blobs are admissible as particles only at the higher threshold levels (e.g. *large* particles), and some only at the lower threshold levels (*e.g. small* particles).

As an alternative to specifying the upper threshold, lower threshold and the number of steps, `[;USLW Threshold type]` allows Dig*Image* to determine the threshold automatically from the intensity histogram of each image as it is processed. In most circumstances better results will be obtained by manual specification of the threshold criteria. However, inexperienced users may find the automatic option easier. The automatic option may also be useful when tracking images whose overall intensity varies with time.

The properties which define a particle are based on its area, shape and intensity distribution. Only blobs which fall within a given range of sizes (based on the pixel area - `[;USLR Lower size limit]` and `[;USLS Upper size limit]` - and linear dimensions - `[;USLX Minimum horizontal size]` and `[;USLY Minimum vertical size]`), shapes (based on the *x*-*y* correlation of pixels satisfying the threshold `[;USLE Ellipticity limit]`) and which have an average intensity sufficiently above the threshold (`[;USLA Minimum average intensity excess]`) will be treated as particles. Within this range there are a number of categories which indicate how confident (or otherwise) we are that this is a (single or multiple) particle. These categories will be elaborated on in the next section.

The location of the particle is determined from the either the centroid of the area satisfying the prescribed threshold, or by the centroid of the volume (area times intensity) satisfying the same threshold. The difference between these two centroids must be less than some predefined limit (`[;USLM Upper limit on centroid mismatch]`) before the blob will be treated as a particle. Which measure of the location will be used in the tracking process is up to the user (`[;USLC Area/Volume centroid]`). The volume centroid is to be preferred as it yields a more accurate position, particularly for particles a few pixels in linear dimension. The accuracy with which the position of the particle is evaluated affects the velocity resolution of the tracking technique. The positional accuracy is a function of the particle size. If the particle is smaller than one pixel, then its position can not be given more accurately than pixel resolution, regardless of which location method is employed. However, under ideal circumstances with a noise-free image and linear response camera, a particle just over one pixel in linear size could be located to an accuracy of 1/256 pixels in that direction. In reality this degree of accuracy is unlikely to be achieved. Nevertheless particles with linear dimensions greater than one pixel, an accuracy of much better than the pixel size may be achieved. Particles should be chosen so that they are as large as possible (in terms of pixels), bearing in mind the assumption that they exactly follow fluid elements (or whatever behaviour is required). One technique for increasing the apparent size of small particles is to defocus the video camera to smear out the particle image. This will be effective, however, only if a very powerful light source were available.

Note also that the greater the use of the available intensity range, the more accurate the volume centroid becomes, provided the image is not saturated.

The options `[;USLZ Count particles in buffer]` and `[;USL0 Suggest thresholds]` are provided to help the user determine appropriate values of the parameters controlling the location of particles. Typically the specification process will start by acquiring a few images of the flow from video tape (not directly from the camera as the intensities tend to be slightly different from those on the video tape). A first *guess* at the thresholds might be made either by looking at the intensity structure of the image using the cursor (you can use `<f5>` to start the cursor menu), from experience and a knowledge of the colour scheme in use, or by selecting `[;USL0 Suggest thresholds]` to provide some idea of appropriate values. The threshold and other controlling parameters may then be modified by an iterative process using `[;USLZ Count particles in buffer]` to determine which particles (and how many) are found. A typical strategy will be to maximise the number of particles found, but simultaneously minimise the number of *blobs* rejected.

2.3 PARTICLE MATCHING

Once all the particles in an image have been found (at $t = t_{n+1}$, say), they need to be related back to the previous image ($t = t_n$, say) to determine which particle image is which physical particle. In Dig*Image* we use a modification of what is known in operations research as the *Transportation Algorithm*. While the problem solved by the transportation algorithm may be represented as a 0-1 totally unimodular integer linear program, it is more efficient and illuminating to take a graph theory approach.

The idea is to choose a set of associations between two sets of entities, such that the set of associations is optimal in the sense that it minimises some linear function of the associations it includes. For the particle tracking, one of the sets is the set of particles **P** at $t = t_n$ and the other the set of particles **Q** at $t = t_{n+1}$. We shall start by assigning a label to all the particles images in the two images. At $t = t_n$ the particle images are labelled p_i for i=1 to i=M, while at $t = t_{n+1}$ they are labelled q_j for j=1 to j=N. We now define a set of association variables α_{ij}. If α_{ij} is equal to one, then we will say that p_i at $t = t_n$ is produced by the same particle as q_j at $t = t_{n+1}$. If α_{ij} is zero, then p_i and q_j represent different physical particles.

For the time being we shall assume that there is one and only one physical particle for each of the particle images. We shall consider groups of particles later in this discussion. For the present it is obvious that, for given p_i, at most only one value of j can give α_{ij} equal to one, otherwise the physical particle must be two places at once! Identical arguments apply for each p_j. If M is equal to N, it may be possible for there to be exactly $M = N$ values of α_{ij} equal to one. However, this will seldom happen in real experiments, where there will normally be fewer than $M = N$ values of α_{ij} equal to one. Moreover, the number of particles images at the two times will not always be equal.

There are many reasons why the number of particles in the image may be different at $t = t_n$ and $t = t_{n+1}$. The simplest is that the particle may have moved outside the region of the flow being tracked, either by moving outside the bounds of the tracking window, or by moving out of the illuminated region (*e.g.* moving out of a sheet of light). To overcome this problem we define α_{0j} and α_{i0} as dummy particles at times $t = t_n$ and $t = t_{n+1}$. Unlike ordinary particles, more than one value of j or i may give a non zero value of α_{0j} or α_{i0} (respectively). In this case a non zero value of α_{i0} indicates that particle p_i at $t = t_n$ has been lost from the image by $t = t_{n+1}$, either by moving out of the image or for some other reason. Similarly, $\alpha_{0j} = 1$ represents a particle q_j present at $t = t_{n+1}$ which was not there at $t = t_n$.

In order to determine the optimal set of non zero α_{ij}, we must first define the functional to be optimised. The only restriction this method puts on the functional is that it is linear in the associations, α_{ij}, and so may be represented by Z, the sum over i and j of $\alpha_{ij}c_{ij}$. Elements of c_{ij} represent the *cost* of associating particle p_i at $t = t_n$ with particle q_j at $t = t_{n+1}$. The optimal solution will be chosen to minimise the *objective* function Z.

Typically the costs c_{ij} will be specified using some function of the particle positions, particle characteristics, temporal history and the physics of the flow. Conceptually the simplest model is to set c_{ij} equal to the separation between particle p_i and particle q_j (c_{0j} and c_{i0} may be set to the distance to the boundaries of the observed region, or the maximum allowable distance a particle may be allowed to travel between t_n and t_{n+1}). The optimal solution will then try to minimise the particle displacements, allowing only associations which do not exceed the cost limits placed by c_{0j} and c_{i0}. The costs c_{ij} could equally as easily be the squares of the displacements, yielding a type of least squares optimal solution.

If we are trying to measure the fluid velocity (rather than Brownian motion, say), then a more appropriate set of cost functions would include some fluid dynamics. This may be achieved at the most basic level by predicting the positions the particles at $t = t_n$ will have at $t = t_{n+1}$ using their velocity (and possibly acceleration) at $t = t_n$. The costs c_{ij} may then be some function of the separation between the predicted position of p_i and the position of q_j. If a particle at $t = t_n$ has only just entered the image, then we are unlikely to have more than a rough estimate for its velocity and so are unable to predict accurately where it might be at $t = t_{n+1}$. To enable matchings to still occur to such particles, we must reduce the costs of associations with them and allow matchings over larger distances than for particles for which we have a velocity history (we may also, however, add some fixed cost for this *new member*). While this may produce some mismatching, the requirement for a much more exact match would not then be satisfied at $t = t_{n+2}$, and so the mismatch would not continue. During subsequent analysis, if we accept only paths which passed through three or more samples during the tracking phase, then we will eliminate any mismatches due to the less stringent matching requirement for a particle with no velocity history.

Additional factors such as the particle size, intensity, shape or even colour may easily be brought into the costing function. Every added component in a well-chosen functional will increase the probability of a correct matching, but at the expense of increased computation. Fortunately, provided the particle seeding density is not too dense, the extra criteria are unlikely to add significantly to the quality of the results. Experience has shown that the tracking results are relatively insensitive to the exact function used for the costs c_{ij}. Any mismatches which arise due to a short coming in the costing procedure will be short lived (they will fail to match on the next step) and may be trapped during the subsequent analysis phase through acceleration checks, for example. DigImage includes information about particle geometry and intensity in a simple multiplicative manner.

For example, suppose the basic cost of an association between particle images p_i and q_j is simply the separation between the predicted position of p_i and the position of q_j, *viz*.

$$B_{ij} = /\mathbf{x}_i + \mathbf{u}_i\delta t - \mathbf{x}_j/, \quad (1)$$

where $\mathbf{x}_i$, $\mathbf{x}_j$ are the positions of p_i at t_n and q_j at t_{n+1} (respectively), $\mathbf{u}_i$ is a measure of the velocity of p_i at t_n, and δt is t_{n+1}-t_n. The cost of the association, c_{ij}, is then related to B_{ij} by

$$c_{ij} = B_{ij}\,\eta_i\varepsilon_j\wp_j\tau_j I_{ij} + N_i + (1 - X_{ij})Y_i \quad (2)$$

where η_i is the new particle discount function, ε_j the ellipticity premium function, $\wp_j$ the size premium function, τ_j the threshold premium function, I_{ij} the background intensity premium function, and N_i the new particle joining fee. We may use additional information about the neighbourhood of the particles p_i and q_j by introducing the cross correlation function X_{ij} between subregions of the image centred on p_i and q_j. The costing function Y_i then determines the contribution this provides to the total cost.

The various weighting functions are defined as follows:

Modifier	Description
η_i	New particle discount. If p_i at t_n does not have any velocity history (*i.e.* it was not matched with any particle at t_{n-1}), then it will be eligible for a discount of *Maximum_matching_distance*/(*New_particle_v_error**δt), where the numerator (set through `[;USPM Maximum matching distance]`) is the maximum distance over which a normal particle association may occur, and *New_particle_v_error* (`[;USM Max velocity error for new paths]`) is the maximum velocity error for a new particle. If p_i at t_n has a velocity history, then N_i is unity and there is no new particle discount.
ε_j	Ellipticity premium. If the ellipticity of p_j falls within specified bounds (set by `[;USLE Ellipticity limit]`), then ε_j takes the value of one. If q_j is outside the bounds, then ε_j will be infinite if q_j is to be ignored. Alternatively, q_j may be considered as two particles, the first of which has $\varepsilon_j = 1$, while the second incurs the premium specified (set by `[;USPE Premium if ellipticity exceeded]`).

$\wp_j$	Size premium. If the size of the particle lies between specified limits (set by `[;USLR Lower size limit]` and `[;USLS Upper size limit]`), then $\wp_j$ is unity. If q_j is too small, then $\wp_j$ is either an inflated value (specified by `[;USPR Premium if particle too small]`) or infinity, depending on whether the particle is to be included or not. If q_j is too large, then it may either be ignored (infinite $\wp_j$), or treated as two particles, the first of which has $\wp_j = 1$ and second of which incurs the premium specified (by `[;USPS Premium if particle too large]`).
τ_j	Threshold premium. As noted earlier, DigImage employs two or more separate thresholds during the particle location phase. Particles q_j which satisfy threshold above the midpoint have $\tau_j = 1$, while those which satisfy only the less stringent (lower) thresholds have τ_j equal to the value specified (in `[;USPI Premium if faint particle]`).
I_{ij}	Background intensity premium. If background intensity pricing is enabled (through `[;USPB Enable background intensity pricing]`) and 511 or fewer particles are being tracked, then if the local background intensity in the neighbourhood of p_i and p_j differ by two levels (the range is divided into eight levels as specified by `[;USLB:  Background intensity evaluation]`), the association suffers a price premium (specified by `[;USPC Premium if two intensity steps]`). If the intensity difference is more than two levels, then the premium is greater (specified by `[;USPD Premium if many intensity steps]`). On the other hand, if the intensity difference is one or less, or background intensity pricing is not enabled, then $I_{ij} = 1$.
N_i	New particle joining fee. If p_i at t_n does not have any velocity history (*i.e.* it was not matched with any particle at t_{n-1}), then in addition to the cost based on the separation between its predicted position and the location where q_j is found, it will also suffer a *new member fee* equal to N_i. Typically the joining fee will be approximately half the maximum cost that may be imposed on the association. If p_i does have a velocity history, then there is no joining fee and N_i is zero. The new particle joining fee is set by `[;USPJ New particle joining fee]`. In most cases the default value of 0.25 is appropriate.
Y_i	Cross correlation costing. This factor multiplies $1 - X_{ij}$ to determine how the cross correlation information is costed in the solution process. Two user-specified costings are implemented, one for particles with no velocity history (set using `[;USPK New particle correlation costing]`) and a separate one for particles with a velocity history (set using `[;USPL Particle correlation costing with history]`). Note that correlation costing is relatively expensive to compute. In most situations it adds little to the accuracy of the tracking process, and so may be disabled by setting Y_i to zero. This is the default value.

Note that this tracking technique is able to cope with flows which have velocity gradients parallel to the viewing direction. As spatial correlation is not used, there is no confusion if a particle at one side of a light sheet (say) is travelling in the opposite direction to one on the other side, even if their paths appear to cross on the video image.

For particles with no velocity history, DigImage will make an estimate of their velocity prior to evaluating B_{ij}. The `[;USN: New particle behaviour]` menu controls how this estimate is made:

Access sequence	Description
`[;USNC Constant predicted velocity]`	Assumes on average the particles are moving with some spatially uniform velocity.
`[;USNL Local velocity estimate]`	Determines the local mean velocity from the other particles and uses this to estimate the velocity for the new particle.
`[;USNR Region-dependent predicted velocity]`	Assumes on average the particles within one region are moving with some spatially uniform velocity, and those in the second region are moving with a different spatially uniform velocity.
`[;USNZ Zero predicted velocity]`	Assumes on average the particles have a zero mean velocity.

The method by which the optimal set of associations is found closely resembles the transportation algorithm. Here we outline the principle of the solution algorithm rather than the specific implementation used by Dig*Image*.

Once the costs c_{ij} have been specified, any value of c_{ij} exceeding c_{0j} or c_{i0} (*i.e.* the cost of a particle leaving the region which is given by the lesser of the cost based on its distance to the boundary and that based on the maximum matching distance `[;USPM Maximum matching distance]`) and the cost is set to infinity: that matching will never occur so we need not consider it. An initial guess is made at α_{ij} such that, for given i = I, j = J is chosen such that c_{IJ} is the minimum of c_{Ij}, and α_{IJ} set to one, provided no value of α_{iJ} is already one. Frequently this initial solution will be very close to optimal, but that will not always be the case. It is now necessary to iterate until the optimal solution is found.

For each iteration, we scan through the list of zero α_{ij} which have finite costs, and evaluate the cost of bringing that association into the solution. Consider the zero association α_{IJ} and suppose that $\alpha_{IJ} = 1$ and $\alpha_{KJ} = 1$. Now if α_{IJ} were to be brought in, then α_{IL} and α_{KJ} must leave the solution, otherwise I and J will be two places at once. Further, as particle images i = K and j = L must be related to some physical particle, it will be necessary to let α_{KL} enter the solution. Thus the *reduced cost* of bringing in α_{IJ} is $c_{IJ}+c_{KL}-c_{IL}-c_{KJ}$. If this reduced cost is negative, then bringing α_{IJ} into the solution is favourable (will decrease the objective function). After scanning through all zero α_{ij} with finite costs, we bring the α_{ij} in which had the most negative reduced cost and start the next iteration. If all α_{ij} have reduced costs greater than or equal to zero, then the optimal solution has been found.

This matching technique is executed twice for every time step in Dig*Image*. The first time, we match $t = t_n$ with $t = t_{n+1}$. Any particles p_i at $t = t_n$ for which we do not find a match at $t = t_{n+1}$ is stored for later use. Particles q_j at $t = t_{n+1}$ for which there was no match at $t = t_n$ are then compared with particles at $t = t_{n-1}$ for which there was no match at $t = t_n$ (provided there was a velocity history at $t = t_{n-1}$). Thus we are able to follow particle paths even if the particle concerned is not visible for one sample. During the subsequent analysis stage care should be exercised in using any data which spans such a disappearance; they may be excluded from such analyses. The main reason behind this second matching phase is to maintain a velocity history for that particle so that its future position may be predicted.

Difficulties with particle paths crossing such that two particles are in the same place at $t = t_{n+1}$ are overcome in a simple manner. We assume that any such conglomeration of particles will have unusual shape and or size characteristics. The combined particles may appear like a very large, or very elliptical (if the two particles are side-by-side) particle. If such an anomalous *particle* is detected, then Dig*Image* will treat it as two particles. The *first* of the pair is treated in the ordinary manner, with the association costs prescribed in the manner outlined above. The *second* will be treated with more caution as it may not be a second particle. The cost of associating it with any particle at $t = t_n$ will be increased by some predefined factor (`[;USPE Premium if ellipticity exceeded]` or `[;USPR Premium if particle too large]`) depending on why the particle is to be considered as two. These premiums are related to the probability that the anomalous blob is infect two particles. At present these factors represent a linear increase in the cost as was outlined above ($\varepsilon_j \wp_j$). In practice these price premiums are seldom invoked during the particle tracking as the relatively sparse particle seeding ensures the probability of two particles coinciding is low. Moreover, if they are invoked incorrectly, then it is unlikely that the path containing the erroneous matching will survive. Note that matching two samples back (*i.e.* to t_{n-1}) is only permitted on the *normal* part of such a particle.

2.4 OPERATION

Prior to tracking an experiment it is necessary to set up a range of parameters which control the tracking process. A complete description is beyond the scope of this document, but they may be classified into six main groups: coordinate system specification, video control, image enhancement, particle location, particle association and data output. Some of these parameters have already been discussed in this document. More complete details may be obtained from the help text within the various submenus involved.

As Dig*Image* utilises the world coordinate system, it is essential that this be defined before the tracking proceeds. This is achieved interactively using the facilities in `[;PWW Determine world coordinate mapping]` (the list should first be initialised by `[;PWV Initialise world coordinate mapping]`) by locating a number of features on an image (typically a grid temporarily placed in the experimental apparatus) and specifying their world coordinates. A variety of mapping functions are then available to fit a mapping between the world and pixel coordinate systems. The simplest function consistent with the optical arrangement should generally be chosen. Typically this will involve a simple linear mapping between pixel and world coordinate spaces.

A second stage of mapping is desirable as there are generally small discrepancies in the image location between one frame and the next due to small synchronisation and timing errors in the video signal. When provided with a set of permanent reference points, Dig*Image* is capable of automatically generating this mapping for each sample it processes. These reference points will typically consist of small points of light on a dark background (or dark dots on a light background) arranged in a vertical strip on either side of the viewed area, just outside the window to be tracked. It is preferable to avoid having particles visible in the immediate vicinity of the points; if they are visible, the points must be clearly distinguishable by a threshold higher than the maximum expected for any particle. Each point must be registered with Dig*Image* (using `[;PRL Locate permanent reference point]`), along with the threshold defining it, and the mapping function selected (by `[;PRM Mapping type for perm cal points]`). At the start of the tracking process, the points previously registered with Dig*Image* will be located automatically, and their current locations used as the reference position (instead of their original registered positions).

For subsequent samples, the new position of the points will be found and a mapping evaluated to transfer any data from the coordinate system in the new frame to the coordinate system in the initial frame before the data is converted to world coordinates. If the root mean square (rms) error in the mapping function exceeds some specified value, then Dig*Image* will force the image to be captured again from video in an attempt to improve the accuracy of the mapping. If after five retries an acceptable rms error value is still not achieved, then Dig*Image* will flag an error, but continue with the tracking process.

Control of the video hardware includes specification of the maximum time over which particles are to be tracked, an the sampling frequency. The tracking may be either forwards or backwards in time (for a flow in which the flows are nearly singular at some point, tracking backwards towards that point may be more effective than tracking forwards from it), with sampling at up to 50Hz (60Hz for NTSC). Note that when sampling at the maximum frequency there will be a small error in the vertical positioning of particles due to the interlacing of the video lines. The vertical resolution will also be reduced from 512 to 256 lines (480 to 240 lines for NTSC). Sampling at this maximum frequency only makes sense when a field integration mode is used. Refer to `Document\Cameras.DOC` for details.

Some of the techniques available for image enhancement by removal of variations in the background illumination have been discussed already. In addition to these a range of filter options are available to clean up the image in a variety of ways, such as low pass filtering. When dealing with high velocities (in pixels per sample), the 1/50 (1/60 for NTSC) time difference between the information contained on the even and odd lines, due to the interlacing of these lines, can lead a particle to appear to be in two places at once. In such circumstances it is necessary to discard half the information present in the image and utilise only the even or the odd lines (not both, unless tracking at the maximum frequency which corresponds to single video field spacing of the samples), and operate with a vertical resolution of 256 pixels. Filters exist for a variety of methods of allowing for the interlace, depending on the velocities concerned (see `[;USI: Filter type]` for more details). Note that if you have a frame integration camera in combination with a mechanical shutter, it is possible to avoid these problems and at the same time achieve a significantly improved resolution. For further details refer to `Document\Cameras.DOC`.

Parameters controlling particle location (in `[;USL: Particle location]`) specify the two limiting thresholds and number of thresholds used in their location, limits on the size and ellipticity of the particle (beyond which it may be treated as two particles), the centroiding technique to be used to define the particle location, and the maximum allowable difference between the two centroids. Some of these location parameters interact with the costing parameters (in `[;USP: Pricing policy]`) which control the association process. The three most fundamental costing parameters specify the form of the costing function to be used (`[;USPP: Policy type]`), the maximum "error" between the predicted position of a particle and its actual position before the two images will be treated as separate physical particles (`[;USPM Maximum matching distance]`), and the maximum velocity error a *new* particle entering the tracking region may have (`[;USM Max velocity error for new paths]`). Some of these parameters are described in more detail in an earlier section. More detailed information may be obtained from the help facility associated with the relevant options.

Results of the tracking process are output both visually and in machine readable form. The visual output provides a number of ways of viewing the particle streaks in addition to diagnostics on the tracking status and cost matrix (controlled by `[;USD Display particle paths]`). Four permanent files are created by Dig*Image*. The first (`base_name.PAR`) stores all the tracking parameters used to control the tracking, while the second (`base_name.WLD`) records details on the coordinate system used. The third file (`base_name.IND`) acts as an index to the main tracking file. Finally the main tracking file (`base_name.PRT`) is a direct access binary file which may be many Mbytes in size. When tracking the maximum number of particles (at present 4095) at 25Hz, over 36MBytes (each sample requires 80 bytes plus 6 bytes per particle in the sample) of data is produced for each minute of experimental time. To operate the particle tracking system effectively it is therefore necessary to have a very large hard disc available, especially if ensemble statistics are required.

3 Subsequent Analysis

Tracking the particles is only a small part of the experimental analysis. It will almost always be necessary to summarise the data obtained from the tracking procedure. This summarisation may take the form of graphical output, or statistics of the flow. The precise nature of the required data will depend on the type of experimental problem being analysed and the aspects of the flow in which the user is interested. Unfortunately it is not possible to produce a single program to analyse the tracking data in exactly the way required for every conceivable use. Instead it will be up to the user to devise suitable analyses for some of the problems they encounter, using the basic tools provided with Dig*Image*.

Among the utilities supplied with all versions of Dig*Image* is a reasonably extensive analysis package called Trk2DVel. The purpose of this package is two-fold: first it provides ready access to a number of the most commonly required analyses, and second it may be used as a template on which further methods of analysis may be developed. For this reason most of the source for Trk2DVel is supplied (in Fortran 77) in addition to all required object modules and an executable version.

The next section, §3.1, details the structure of the `.IND` and `.PRT` tracking files, while the following section, §3.2, outlines the main routines which should be used to obtain velocity-position-time data from the files. Further, more detailed information may be found in `Document\Trk2Code.DOC`.

3.1 STRUCTURE OF FILES

As noted in section 2.4, particle tracking under Dig*Image* produces four output files. The first, `base_name.PAR`records the various parameters used to control the tracking process, some of which will be needed during the analysis phase. The pixel to world coordinate system mappings stored in `base_name.WLD` will not generally be needed. The two main tracking files are `base_name.IND` and `base_name.PRT`. The first of these is a formatted file to act primarily as an index to the `.PRT` file. In addition to this indexing role, the `.IND` file contains information relating to the status of the tracking at each time step. Each line in the file contains the following information:

```
iSample iStartRecord iEndRecord rmsError nMatches iUseBuffer iReDo
    iTimeCodeCorrection
```

where

Data Name	Data Type	Description
`iSample`	`INTEGER*2.`	The sample number of the information contained in this line. The time for this information is the product if iSample and the time step size set by `[;USTS Sample spacing]`.
`iStartRecord`	`INTEGER*4.`	Acts as an index to the record in the `.PRT` file containing information about the first particle in the current sample.

iEndRecord	INTEGER*4.	Acts as an index to the record in the .PRT file containing information about the last particle in the current sample. The number of particles for the current sample is therefore iEndRecord-iStartRecord+1.
rmsError	REAL*4.	Records the root mean square (rms) error in the permanent reference point mapping function for the image associated with this sample. Note that if no reference points are used this field will contain a value of -1.0. If too few reference points were found to generate the mapping, this field will contain a value of 999.0.
nMatches	INTEGER*2.	Indicates the number of successful normal matches between particle images at this and the previous time step.
iUseBuffer	INTEGER*2.	Indicates which buffer was used to store the image used for this time step.
iReDo	INTEGER*2.	Used to indicate the number of repeat attempts **Dig*Image*** had at acquiring an image with a satisfactory rms error for the reference point mapping. A value of zero indicates that the image was of adequate quality on the first attempt, while a value of -1 indicates that **Dig*Image*** was unable to acquire a suitable image even after five attempts (the image will be accepted after five attempts, even if it is still of inadequate quality).
iTimeCodeCorrection	INTEGER*2.	This entry is used to store any corrections or errors which result from the time code pulses on the audio track of the video tape showing an inconsistent count on the video field strobe (see the help facility in the [;VV: Time code audio pulses] menu for more details). The following values are defined: 0 No correction needed. 2 Two field strobe pulses lost but corrected for. >2 More than two field strobe pulses lost but only 2 corrected for. -1 or -2 One or two interrupts lost, but corrected for. <-2 More than two interrupts lost - only two corrected for. 100 Time code offset re-zeroed. 101 Time code pulse not found. If the tracking is undertaken from either a JVC BRS822 video recorder, or from images stored on the hard disk, then this field does not contain any useful information.

The last two entries may be used to discriminate between high and low quality tracking data. The option exists within Trk2DVel to exclude paths which have anomalous values for either of these entries.

The `.PRT` file is a direct access binary file. Each record within this file consists of six bytes containing information about a single particle at a single time step. The information stored depends on the maximum number of particles being tracked. For 511 or fewer particles (tracking started by `[;UG Start particle tracking - 511 particles]`):

Byte(s)	Bits	Data Name
0	all	`iFrom`
1	0	`iFrom`
1	1-3	`iFlags`
1	4-7	`iType`
2&3	all	`iXFile`
4&5	all	`iYFile`

For 4095 or fewer particles (started by `[;UH Start particle tracking -4095 particles]`) the `.PRT` file contains:

Byte(s)	Bits	Data Name
0	all	`iFrom`
1	0-3	`iFrom`
1	4-7	`iType`
2&3	all	`iXFile`
4&5	all	`iYFile`

In both cases

Data Name	Description
`iFrom`	Nine (511 particles) or twelve (4095 particles) bits - unsigned integer. If `iFrom` >= 0, then `iFrom` is index the particle had at the previous time step. At each time step the particles are numbered from 0 to n-1, where n is the number of particles at that time step, such that the particle number at a given time step corresponds to its offset from `iStartRecord` into the `.PRT` file. The `iFrom` field gives the offset that the current particle had at the previous time step. A value of 511 (or -1) indicates that the particle had no match at the previous time step.
`iFlags`	Three bits (511 particles only). The `iFlags` bits may be used for a number of purposes. In the present version they are used to store the local background intensity or region (see section 2.1).

`iType`	Four bits - 2's compliment integer. The `iType` bits store the type of particle the current record corresponds to. The `iType` values are as follows: 0 Normal particle. 1 Elliptical particle - first of pair. -1 Elliptical particle - second of pair. 2 Large particle - first of pair. -2 Large particle - second of pair. 3 Small particle - horizontal or vertical extent < minimum, but at least 2 pixels. -3 Small particle - area < minimum. 5 Faint particle 6 Faint elliptical particle - first of pair. -6 Faint elliptical particle - second of pair. 7 Particle matched two samples back.
`iXFile`	Two bytes - unsigned integer. Rescaled x coordinate. To economise on file storage, the world coordinates are rescaled from the `REAL*4` form used internally by Dig*Image*, to a two byte unsigned integer value for storage in the `.PRT` file. This rescaling may be expressed as `iXFile = (x - xMin)*65535/(xMax-xMin)` where `iXFile` is the value stored in the file, `xMin` is the minimum value the `x` coordinate may take, and `xMax` is the maximum value for the `x` coordinate.
`iYFile`	Two bytes - unsigned integer. Rescaled y coordinate. To economise on file storage, the world coordinates are rescaled from the `REAL*4` form used internally by Dig*Image*, to a two byte unsigned integer value for storage in the `.PRT` file. This rescaling may be expressed as `iYFile = (y - yMin)*65535/(yMax-yMin)` where `iYFile` is the value stored in the file, `yMin` is the minimum value the `y` coordinate may take, and `yMax` is the maximum value for the `y` coordinate.

Note that a suite of Fortran linkable assembler subroutines is available to undertake the conversions between the file format and a valid Fortran format see `Document\Trk2Code.DOC` for details.

3.2 BASIC OPERATION OF TRK2DVEL

The Trk2DVel system consists of a suite of three programs, all of which are accessed by the `Trk2DVel` command. User-written programs may readily be added to the suite in a user-transparent manner. The structure and operation of Trk2DVel is similar to that of the main Dig*Image* suite, though its purpose is to produce numerical and graphical analyses of the particle paths, rather than to acquire analyse images. As a result it does not include the basic acquisition and image manipulation facilities of the main Dig*Image* suite.

As with the main Dig*Image* system, the major part of the user documentation concerning Trk2DVel is provided by the context sensitive Help System accessed through `<f1>`. Trk2DVel may be run either interactively, or using command files with the same basic file format and syntax as Dig*Image*.

Only a smaller range of the function key short cuts are available, reflecting the less general nature of Trk2DVel. In particular the following Dig*Image* keys are implemented:

Function Key	Description
`<f1>`	Give help on the current activity. Pressing this key in response to a submenu will bring up a help page giving details of the various options available within the current submenu.
`<shift><f1>`	This key combination will display this General Help file.
`<ctrl><f1>`	This combination brings up a brief summary of the facilities available through the function keys.
`<f2>`	Displays a listing of the files in a user-specified directory. Defaults to the current directory.
`<shift><f2>`	No effect from `->` prompts. Displays the history list for `=>` prompts.
`<ctrl><f2>`	Change the current working directory.
`<f3>`	Display the previous output buffer.
`<shift><f3>`	Display the next output buffer.
`<ctrl><f3>`	Prompt for a buffer to be displayed.
`<f4>`	Same as `<ctrl><f4>`.
`<shift><f4>`	Same as `<ctrl><f4>`.
`<ctrl><f4>`	Specify a particular output look up table.
`<alt><f4>`	Show the DigImage news file (`HlpFiles\@News.HLP`).
`<f6>`	Open a DigImage Journal File. All keyboard input during the DigImage session (unless a command file is subsequently opened) will be recorded in this file along with comments, warning messages etc. This facility is of particular value for use when creating command files. The Journal File may be edited to provide the desired command sequence, or, for relatively simple command structures, used in its raw form. By default a `.JNL` extension will be added to the file name if none is specified.
`<shift><f6>`	Close the DigImage Journal File previously opened by `<f6>`.
`<ctrl><f6>`	Type on the screen the contents of the specified file, one page at a time.
`<alt><f6>`	Delete the specified file.
`<f11>`	Copy current buffer to user-specified destination.
`<ctrl><f11>`	This function key can be used to start the current screen saver at any stage where user input is requested. Note that if the frame grabber is in active use (either acquiring images or performing ALU operations) then only the computer monitor screen saver will be used.
`<alt><f11>`	The normal automatic starting of the screen saver can be enabled or disabled by this key combination. The screen saver will be automatically invoked under the following conditions (only the first three apply for the computer monitor): The screen saver is not disabled Trk2DVel is in a menu The specified time must have elapsed since the last user operation (this time is set by the `CONFIGUR` utility) The frame grabber is not acquiring an image The frame grabber is not be executing an ALU instruction

3.3 CONFIGURATION OF TRK2DVEL

Trk2DVel should not normally be entered within a given directory until after at least one experiment has been tracked (or copied to) that directory. Trk2DVel expects the file `Track2D.DIG` (used to store the controlling parameters for particle tracking) to exist in the directory. If it does not, Trk2DVel will generate an error message to this effect.

In the same manner as DigImage generates the status file `Status.DIG` to store the operational status of DigImage, Trk2DVel creates the status file `Trk2DVel.DIG`. The first time Trk2DVel is started from a given directory it will be unable to find the `Trk2DVel.DIG` file and will generate the message

```
WARNING:
Using default path settings - use [;C: Configure Trk2DVel] to
change
Press any key to continue...
```

This warning message indicates one of the primary functions of `Trk2DVel.DIG` will use default values. In particular `Trk2DVel.DIG` stores information on how the particle path information is to be used to evaluate the particle velocities. Trk2DVel can be configured to utilise the path information in a number of different ways using `[;C: Configure Trk2DVel]`, the chosen configuration then being stored in `Trk2DVel.DIG` along with other status information. The configuration menu has the following options:

```
                    Configure Trk2DVel

A  Minimum acceptable path length   :
C  Use large particles              :
D  Use small particles              :
E  Use elliptical particles         :
F  Use faint particles              :
I: Invalid samples - restart criteria
P  Use first matching in a path     :
R: Local intensity regions          :
T  Use particles matched two samples back:
U: Type of calculation for velocity :
V  Period for calculating velocity  :
Z  Set command file search path
Q  Return to parent menu
->
```

Central to the evaluation of velocity from the particle paths are options `[;CA Minimum acceptable path length]`, `[;CU: Type of calculation for velocity]` and `[;CV Period for calculating velocity]`. The first of these specifies the minimum length a path must be before it may be considered valid, while the third gives the length within this path to be used for calculating the velocity. The minimum length should always be greater than or equal to the velocity period; in most cases the two will be set equal. Choosing the period for the velocity is a compromise between velocity resolution, the velocity time scales and the average length of time a particle remains in the light sheet. The larger the number of time steps which may be used to calculate the velocity the more accurately the velocity may be calculated, provided the velocity has not changed significantly during that period. The period should always be more than two time steps to ensure invalid new particle matchings do not contaminate the results. (Note: times may be specified in the minutes:seconds and ::fields format of DigImage, and as ;samples giving a period of *samples*time_step*). Typically a velocity period of four or five time steps (*i.e.* ;4 or ;5) provides an acceptable balance between the various criteria. As a check the same paths may be processed using different velocity periods and their results compared.

The menu produced by `[U: Type of calculation for velocity]` determines how the positions of the particles falling within the velocity period are used to determine the velocity for the mid-point of the path:

```
                Type of Calculation for Velocity

C  Chebeshev linear fit
L  Linear least squares
M  Quadratic least squares
P  Extreme points
Q  Return to parent menu
->
```

With the exception of `[P Extreme points]`, all these options use information about the particle positions throughout the `[V Period for calculating velocity]` (`[P Extreme points]` simply performs a finite difference calculation on the extremes of this period). In general the least squares approach is to be preferred with `[L Linear least squares]` evaluating linear fits to the x - t and y - t data within the velocity period, while `[M Quadratic least squares]` fits a quadratic polynomial to the same data. For good quality data where interlace filtering has not been necessary, there is little to choose between the linear and quadratic fits to the same number of time steps. However, if the average path length is sufficiently long, then the quadratic fit may be used over longer periods, while the linear fit is less prone to noise with lower quality data. A comparison of the velocity probability density functions for the two approaches generally highlights any differences.

In addition to determining how the particle data is used to determine the velocities, the `[;C: Configure Trk2DVel]` menu also specifies which particles and paths may be used. Options `[;CC Use large particles]`, `[;CD Use small particles]`, `[;CE Use elliptical particles]` and `[;CF Use faint particles]` may be used to deselect certain categories of particles such that a path will be considered to terminate when the particle falls in one of the deselected categories. For example, if `[;CF Use faint particles]` is set to `No`, and on a path twenty time steps long the particle image at the fifth time step on a path was characterised as *faint* (*i.e.* it satisfied only the lower band of thresholds during the tracking phase), then the path would be considered as two paths, one extending up to the fourth time step, and the other from the sixth to twentieth time steps. In general the velocity structure of a well set up experiment should not be sensitive to the settings of these options. They will affect the number of particle paths extending over at least a given number of time steps, but should not affect the measured velocity probability distribution.

While tracking, Dig*Image* performs two stages of matching. The first is between the new time step and the previous time step, and the second between those particles not matched in the first stage and those not matched two time steps back. Depending on the particle density, there may be around 10% additional matches in this second stage. The option `[;CT Use particles matched two samples back]` allows paths including such a matching to be deselected. If such paths are included, then a simple linear interpolation is used to estimate the position of the particle at the time step where it was not located.

In rare cases when using low quality data (typically data tracked with `[;USPM Maximum matching distance]` set too large) it may be desirable to exclude the first matching in a path. This may be achieved through `[;CP Use first matching in a path]`.

During the particle tracking DigImage records a number of events which may affect the quality of the tracking data. Subsequently during the analysis phase, it may be desirable to reject data associated with these events by stopping all the particle paths when one of these events occurs. The menu `[;CI: Invalid samples - restart criteria]` determines which events force the particle paths to be restarted. The events concern either the root mean square error in the reference point mapping (a large rms error indicates a low quality image), or the time code pulses placed on one of the audio channels (used to trap errors in the VTR tape position). The options in the menu are:

```
                    Invalid Sample Restart Criteria

R  Retry for reference point rms error        :
S  Reference point rms error not satisfied     :
T  Unable to find time code pulses :
U  Time code pulse error >  2      :
V  Time code pulse error < -2      :
Z  Time code pulses re-zeroed      :
Q  Return to parent menu
->
```

The first two options involve the rms error in the reference mapping. `[;CIR Retry for reference point rms error]` is not normally a serious event. If during the tracking phase DigImage detects an rms error larger than the limit set by `[;USZ Limit on rms error for mapping]` then it will try to improve the quality of the image by repeating the acquisition of the frame. In general the quality will be improved on the first retry and there is no need to restart the particle paths during the analysis. However, if after five retries the rms error is still too large, then DigImage will continue tracking, noting that it could not satisfy the requirement. Under most circumstances samples flagged with this condition will be of unacceptable quality, and so the particle paths should be restarted. This is achieved by setting `[;CIS Reference point rms error not satisfied]` accordingly.

The other four options in this menu concern the time code pulses on the audio channel. Time code pulses must be added to the video tape by the DigImage option `[;VVA Add time code pulses]` prior to tracking the particles. The time code pulses are used to detect and correct any errors in the tape transport due to the limitations of the video hardware. If the time code pulses are not present, or there is something wrong with them, then DigImage may not be able to reconstruct the time sequence with the required 100% accuracy. `[;CIT Unable to find time code pulses]` determines whether or not the particle paths are to be restarted if DigImage was unable to locate the pulses while tracking the particles - normally this event should restart the particle paths.

The time code pulses are designed to correct errors of between +4 and -2 video fields (+2 and -1 frames). However, under normal operation the errors should be confined between +2 and -2 fields. An error detected outside this range normally indicates a more serious timing problem, and so should be used to restart the particle paths. Similarly under some (very rare) circumstances the error is such that DigImage is not sure whether the tape is behind or ahead the position it thinks it should be in. The strategy here is that DigImage will re-zero the time code mechanism, effectively forcing the current situation to be correct. `[;CIZ Time code pulses rezeroed]` should normally be set to restart the particle paths should this event be detected.

Another option in `[;C Configure Trk2DVel]` is `[;CR: Local intensity regions]` which produces a menu controlling the use and meaning of the different local intensities saved along with the particles under some combinations of the controlling tracking parameters.

```
                    Background Intensity

I  Included regions              :
0  Concentration for region 0    :
1  Concentration for region 1    :
2  Concentration for region 2    :
3  Concentration for region 3    :
4  Concentration for region 4    :
5  Concentration for region 5    :
6  Concentration for region 6    :
7  Concentration for region 7    :
Q  Return to parent menu
->
```

`[;CRI Included regions]` determines which of the intensity regions are to be included in the paths used. If a path contains a particle image with a local background intensity falling into one of the levels which is not included using this option, then the path will be considered as stopping before the corresponding time step (and possibly a new path starting at the next time step).

Some of the facilities in Trk2DVel are able to operate either on velocity or concentration flux. For concentration flux measurements, options `[;CR0 Concentration for region 0]` to `[;CR7 Concentration for region 7]` should be set up with the correspondence between the eight intensity regions and the concentration of some scalar field. The required calibration procedure is beyond the scope of this document.

The final option in `[;C Configure Trk2DVel]` is `[;CZ Set command file search path]`. This option allows the specification of an optional search path for command files when started by the `!P` directive (either interactively or from within another command file). If a requested command file is not found in the current directory, then the path specified here will be searched. If the file is still not found, the `%DigImage%\Macros` directory will be searched.

3.4 FACILITIES

In this section we briefly list the main facilities available in the three basic Trk2DVel modules. It is not necessary for the user to know in which module a given feature resides as Trk2DVel automatically changes to the required module. The main menu is reproduced below:

```
               Track 2D Velocity Analysis Menu

A  Lagrangian autocorrelation functions
C: Configure Trk2DVel
D  Probability density functions
E  Create file describing ensemble mean
F  Produce film of grid file
G: Produce an individual grid file
H  Produce power spectra of grid file
I  Information on tracking parameters
J  Joint probability density functions

O  Particle dots
P  Particle paths
```

```
S  Spatial two point correlation functions
T  Create file describing temporal mean
V  Evaluate basic velocity statistics

Q  Exit Trk2DVel
->
```

The fundamentally Lagrangian nature of particle tracking is exploited in `[;A Lagrangian autocorrelation functions]`. The basic data for these functions is pairs of velocities evaluated at different times on the same particle path. The correlation is averaged over all particle paths in a specified interval, and expressed in terms of the temporal separation between the two velocity evaluations. The tracking region may optionally be subdivided into a source region in which all the paths must start, and a continuation region (containing the source region) in which the particles must remain in order to contribute to the autocorrelation functions. The configuration menu `[;C: Configure Trk2DVel]` was covered by the previous subsection.

Velocity and acceleration probability density functions may be evaluated using `[;D Probability density functions]`. These functions may be expressed as a function of both velocity (or acceleration) and time, and may be confined to particles in a window smaller than the tracking window if required. The window may be either rectangular, or specified using a mask buffer. The probability density functions are particularly useful as the basis of a comparison between the different parameters determining the method of evaluating the velocity set in `[;C: Configure Trk2DVel]`.

The `[;E Create file describing ensemble mean]` option is designed specifically for taking an ensemble of experiments and determining the spatially and temporally varying mean motion within that ensemble. It operates by gridding the particle velocity data onto a 21x21 grid and averaging this gridded data (suitably weighted by the particle densities) over the different realisations within the ensemble. As an option a degree of temporal averaging may also be included in this computation. The file created by this option may then be used with many of the options in Trk2DVel to attempt to remove the effect of the spatially and temporally varying ensemble mean motion from the basic velocity statistics.

Animations of gridded velocity data produced by `[;E Create file describing ensemble mean]`, `[;G: Produce individual grid file]` and `[;T Create file describing temporal mean]` may be displayed through the `[;F Produce film of grid]`. In addition to displaying the velocity data as a grid of arrows, a single scalar field (such as vorticity, divergence, stream function etc.) may simultaneously be displayed as false colour. As an option the sequence may be animated onto video to produce a film comparable with those provided by numerical simulations.

The most general method of gridding the particle velocity data is provided by `[;G: Produce individual grid file]`. Grids with up to 64x32 mesh points may be produced with the option of a moving average through time to act as a low pass temporal filter. A number of gridding procedures are supported, based on a weighted least squares approach. In general the best results are obtained with relatively high particle densities (in excess of 1000 paths being used for velocity determination). With such densities the simplest strategy of fitting a constant to the velocity using the weighted least squares is both the most robust and the fastest (by a considerable margin). However, the higher order fits are able to utilise the data with less spatial filtering and obtain velocity gradients directly from the fitting process, rather than needing to finite difference the velocity grid.

If the grid file produced by [;G: Produce individual grid file] has a resolution in both directions corresponding to an integral power of two, then [;H Produce power spectra of grid file] may be used to produce approximate power spectra of a single or (preferably) an ensemble of grid files. At present this facility is of very limited value as the grid files are confined to less than two decades of scales. However, by considering an ensemble of experiments focusing on a hierarchy of different scales (by zooming in on the experiment), it should be possible to construct more complete spectra using appropriate matching conditions on overlapping regions of the wave number scales.

The option [;I Information on tracking parameters] is designed to produce a simple formatted file listing the tracking parameters in force while analysing a given experiment.

The option [;D Probability density functions] evaluated separate pdfs for the two measured velocity components, whereas [;J Joint probability density functions] is designed to determine the joint pdfs. The data is displayed as a sequence of three-dimensional surfaces with the horizontal coordinates corresponding to the two velocity components, and the vertical scale giving the probability (frequency). Temporal averaging may be employed and/or time series generated with optional animation onto video tape.

The most basic form of output from Trk2DVel is plotting particle paths using [;O Particle dots] and [;P Particle paths]. The paths may be represented as dots (either option), or lines joining the successive positions of the particles ([;P Particle paths] only). With [;P Particle paths] the paths may be restricted to only those satisfying certain length constraints (set locally in the option) and/or the type constraints (as set by [;C: Configure Trk2DVel]). The paths may be animated onto video tape with time expansion or compression as desired.

Two point Eulerian velocity correlation functions may be evaluated by [;S Spatial two point correlation functions] in a wide variety of circumstances. The data used is binned and correlated directly from the raw particle velocity data rather than introducing any errors through using gridded velocity data. If temporal or ensemble averaging is appropriate then high quality statistics may be gathered even when using relatively low particle densities. As with many of Trk2DVel's options, the statistics may be restricted to particles falling within a given more restricted region than the tracking window. The restricted region may be either a rectangular window, or a more general region specified by a mask buffer. The results are presented graphically as either a sequence of false colour maps of the correlation functions, or three dimensional perspective plots. Integral length scales are also calculated.

The option [;T Create file describing temporal mean] is very similar to [;E Create file describing ensemble mean] in that it produces a description of the mean motion on a 21x21 grid which may be used by many of the other Trk2DVel options to correct statistics for the spatial structure of this temporal mean motion.

The most basic mean and fluctuation velocity statistics are produced and plotted by [;V Evaluate basic velocity statistics]. While less sophisticated than the data provided by many of the other options, these basic statistics are the most readily understood.

For further details on any of these options, consult the context sensitive help system while executing the option (*i.e.* press <f1> after starting the option the option may be terminated subsequently by pressing <escape>).

4 Advanced Features

The particle tracking may be customised in a number of ways with user supplied subroutines to perform pre-processing of the images and to determine and record auxiliary particle-tagged data such as particle size, shape and orientation.

4.1 PARTICLE-TAGGED DATA

The particle tracking process includes a user-hook to enable additional data concerning the particles being tracked (or another feature of the flow) to be stored in a special file with the `.HOK` extension. The user-hook is contained in the Fortran file `User_Obj\Trk2Hook.OBJ`. This hook is provided full access to the raw image, along with a list of the particles already found and their associated intensities. One record of data must be saved for each particle in the list. This particle-tagged data may be of any length or complexity, depending on the user specification. Typical uses of this facility are to record size/shape/orientation of the particles, or features about the background intensity in the neighbourhood of the particle. For complete details, please refer to `Document\Trk2Hook.DOC`.

4.2 IMAGE PREPROCESSING

This feature is also invoked through the `User_Obj\Trk2Hook.OBJ` file and may be used to perform arbitrary pre-processing operations on the images prior to locating the particles. Again, refer to `Document\Trk2Hook.DOC` for details.

5 Experimental Hints

In this section we list a number of suggestions which you may find useful when setting up and performing experiments with particle tracking.

Black book covering film stuck to the inside of the tank produces a much blacker background (and hence better contrast) than simply turning the lights off or covering the outside of the tank with black material or paper. The film is easy to remove later, and helps protect the tank from damage.

When mixing Pliolite particles with water, place the particles in a clean beaker then add sufficient photographic wetting agent to make the particles appear “wet”. Leave them in this state for about 30 minutes before carefully adding enough fresh water (about 10 times the volume of particles) to allow the particles to be suspended. This solution may then be added to your experiment.

To eliminate any fine dust mixed in with the samples, mix up the Pliolite then, after the larger particles have settled, pour off most of the water. Add more water (and, if necessary, more wetting agent) and repeat this process until most of the dust is removed.

Pliolite left standing in fresh water absorbs some of the water, reducing the bulk density of the particles. This process may be accelerated by boiling the particles in water. Producing particles with a range of densities is often important when measuring stratified systems. By mixing particles which have been treated in various different ways it is possible to increase the vertical spread of particles in the water column.

The colour and density of the particles may be changed by painting them. We have had considerable success using acrylic paint from art shops. The pain has a slightly larger density than the particles. As it is not possible to add exactly the same volume of paint to every particle, the net effect is to increase the range of densities, a feature very useful for continuously stratified system. A second reason to paint the particles is to allow the identification of different sets of particles based on their colour. These sets may then be distinguished by introducing a suitable colour filter in front of the camera. These filters may be attached to a mechanical shutter to allow interlacing of images of the different sets of particles.

A step by step tutorial is also provided in the file `DigImage\Macros\Track.CMD`. This macro may be used either with the sample tracking movie available from `http://tiki.damtp.cam.ac.uk/digimage/examples/tracking/`, or by printing it out as step by step instructions on how to set up **Dig*Image*** for particle tracking.

Index