3D GPR Data

Collection
Interpolation
Processing
Migration
Interpretation

3D GPR Data

Collection
Interpolation
Processing
Migration
Interpretation

impulse RADAR

Data Collection with Modern 3D GPR-Arrays

People new to GPR-Arrays may think that array-data means more of the same thing. While this is partly true, there are other details, which, if considered, may ease the following stages of loading, processing, and interpretation of such data. This note aims to give a few hints on the data collection process and the subsequent management of collected data.

Figure 1, Typical environment in which GPR-Array data is gathered

Data volume & density:

A small 3D project can hold some 5 GB of raw data in total. Not a big file by modern standards and easy enough to transfer using a memory stick. However, from a data security and processing perspective, it's not wise to put all that data into a single file. Why? Cheap memory sticks are prone to file corruption during transfer, and this can be especially problematic if the data is in a single file and affected in any way not immediately noticeable by the operator.

Consequently, we recommend dividing even small projects into several parallel swaths (if possible). Also, the data volume is linear to the point distance, where half the point distance means double the data volume. It is usually of no benefit to collect data with a higher density than half the array channel spacing. Therefore, an array with 8 cm channel spacing equates to a point distance of 4 cm.

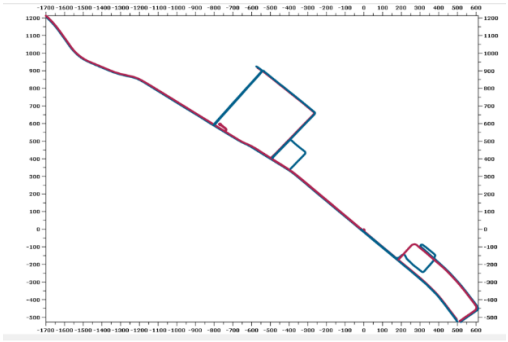

Figure 2, A small project, with respect to raw radar data, but note; the square holding the data is about

Navigation (in the data):

Figure 2 above shows a rather small project, as far as the raw radar data is concerned (approx. 4.5 GB), but the interpretation of a project like this means navigating from a km-scale down to a few meters, which puts quite a high demand on the processing software in use.

Positioning:

Positioning is, by far, one of the biggest talking points concerning the collection of array data. The use of high precision RTK-GPS is the most convenient and efficient positioning method and, for those reasons, takes preference over the use of total stations. However, this convenience becomes ineffective in areas where the signal is interrupted, e.g., by tree cover, tall buildings, or other overhead obstacles. Ultimately, it is the survey environment itself that determines the positioning method to use. For the descriptions that follow, it's important to note that only RTK-fix is sufficient and that any loss of RTK-fix will cause extra work in the data management.

Sometimes, it's possible to salvage a project with poor positioning; however, if the project is large with a high percentage of positioning errors, it may be more economical to re-survey with better positioning than to expend valuable time trying to fix it. Another important consideration that may later ease processing is the choice of positioning density during data collection, since too high a density may

cause extra work. If you don't have adequate control over the positioning process, it makes little sense to deploy to the field for data collection.

Sharp turns during data collection:

When collecting data in the field, it's perfectly feasible to move the array in a manner that produces a sharp turn, or radius, along the collected swath. However, it's a good idea to think about what this kind of maneuver will do to the subsequent data management. As illustrated in Figure 3, such a radius results in data that is significantly stretched along the outer perimeter, while being compressed along the inner perimeter. How this may affect the final image depends on how the data was collected. For example, if the point distance is set to 4 cm during data collection and 8 cm during interpolation, then it may work, but interpolating to the same bin-size as the point distance will definitively impact the data. Therefore, it's important to consider such factors when planning a survey; if such turns are unavoidable, structure the survey so that data collected at these points is not the most important to the overall survey.

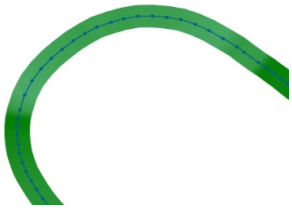

Figure 3, Sharp turns should at best be avoided during data collection

Holes in data

What happens with areas not covered by radar data? Well, modern software offers some ability to interpolate data into such empty spaces, but sometimes applying regularization may be a better choice. Regardless of the theoretical function employed, if the empty spaces are too big, no software can fix it, and those areas will be useless for interpretation. Another less obvious issue is that the interpolation/ binning function takes up memory space on the processing computer, but to what extent is dependent on the chosen processing software. A project like that shown in **Error! Reference source not found.** may be difficult to process due to the very large, unfilled, and closed areas. Of course, if opting to process by manually defining the areas to interpolate ('chunking'), it may always be possible, but that's rather old fashion.

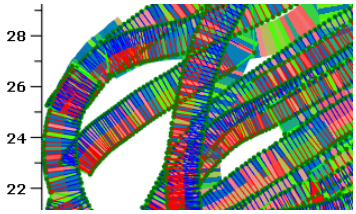

Figure 4, Project with areas not covered by radar data

Figure 5 shows a project with 4.5 GB of raw radar data, the same as the project shown above in Figure 2. However, the layout of this project is much better concerning data management, because it's easy to navigate, has no sharp turns, nor holes in data.

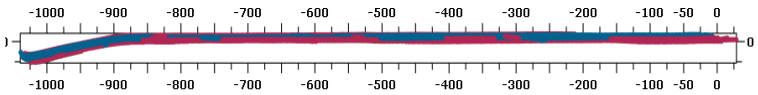

Figure 5, This project is of the same size, with regards to raw radar data as the project in figure 2, 4.5GB. However, this one will be easy to navigate in as well as having no artefacts from polarizations or turns.

Takeaway

Even with the best planning, a real-world project could contain data that is less than optimal. Therefore, when dealing with the data volumes from a modern GPR-array system, it's always advisable to get rid of problematic data as early as possible. The next note in this series will deal with the topic of data QA/ QC and will discuss useful tools for the selection and management of data imports.

Geometry Clean Up and QA/QC of Raptor Data

The introductory note in this series offered advice on the collection of GPR array data. The key takeaway of which was that a real-world project could contain data that is less than optimal. Therefore, this note focuses on the topic of data QA/ QC and will discuss useful tools for the selection and management of data imports into processing software. The primary objective being the import of quality data and efficient workflows.

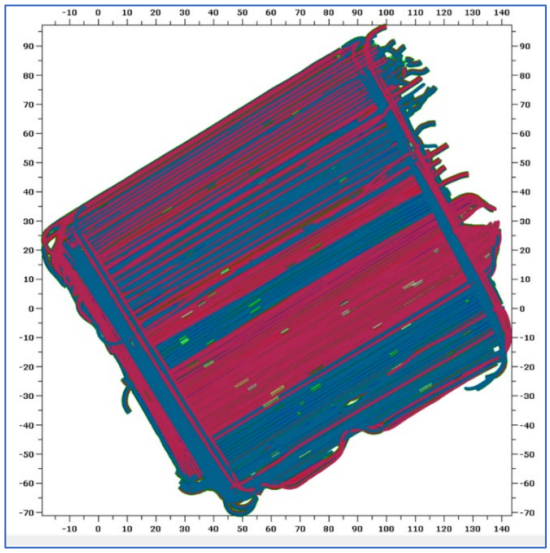

Figure 6, A project, with 175 input files, containing 2800 individual radar profiles

Figure 1 shows a project containing 2800 individual GPR-profiles, which combine to form 175 input files for easier management, although, still a substantial number to handle.

Further, the original data contains over 70000 positioning points, a large percentage of which are problematic — consequently, data sets such as these require practical tools to sort out problems early on before processing.

Figure 2 shows a close up of one part of this project, where the zoom function reveals clear positioning errors (self-intersecting swaths) as well as data swaths that don't make sense.

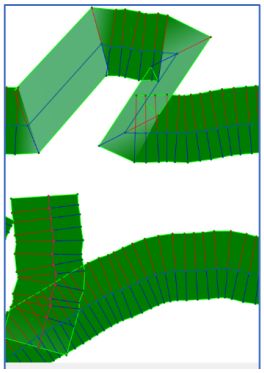

Figure 7, Zoomed picture revealing erroneous data

Swath statistics

Figure 3 shows the swath statistics tool, which is a useful first step to identify essential data readings outside the project norms. In the example shown, the average position density is approx. 3-4/m, but some read as low as 0.06/m. It is safe to assume that these swaths will cause a problem if they import, so uncheck to ignore.

Another noticeable variation is in swath length, where some files indicate only a few meters versus an average of 125 m; again, a simple uncheck of the problem files will omit them from import.

	Swath	Points/m	Traces/m	Start time	End time	Total length, m
☑ 58	ALDERSHOT_...	3.78	20.78	11:21	11:21	105.002
☑ 59	ALDERSHOT_...	0.06	0.16	11:39	11:40	88.172
☐ 60	ALDERSHOT_...	0.06	0.08	11:48	11:48	84.096
☐ 61	ALDERSHOT_...	0.06	0.08	11:50	11:50	95.826
☑ 62	ALDERSHOT_...	4.43	20.72	08:31	08:32	131.700
☑ 63	ALDERSHOT_...	3.20	21.33	08:32	08:33	121.351

Figure 8, Statistics showing positioning and data density and total length of profiles

Colour coding statistics

Colour coding is a simple way to highlight swaths with problematic positioning density. Easy to identify visually, a simple cursor mouse-over reveals specific information concerning the swath file name and position, as per the example in Figure 4.The density of radar data may be treated in a similar way to highlight problems with the odometer values and wheel slip.

Figure 9, Identifying outliers by colour coding

Removal of positioning data

Continuing with the same project example, a lot of data on the perimeter will not process well in 3D, so removing such points will speed up the data processing and reduce the amount of PC storage space required. Figure 5 shows an example of a simple tool to mark and remove such positioning points.

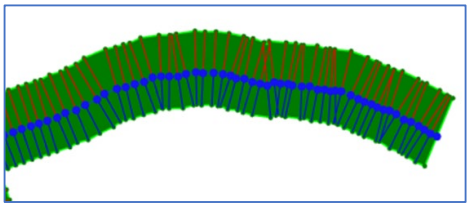

Figure 10, means of effectively mark and delete positioning points, may significantly reduce workload in later stages

Reducing the positioning density

As indicated earlier, swaths with a very high density of positioning will be problematic upon import, which is due to the self-intersection of swaths, an effect typically caused by the GPS antenna swaying side-to-side. Reducing the positioning density by half from 3-4/m to 1-2/m will decrease this problem. Although it does mean removing some data from the project, it will simplify and speed up the processing time.

Final clean-up and radar data import

Even after observing the preceding steps, there may still be some cause for errors in the data. Modern processing software should be able to warn the user of this and guide them on where to search for such errors, as per the example shown in Figure 6.

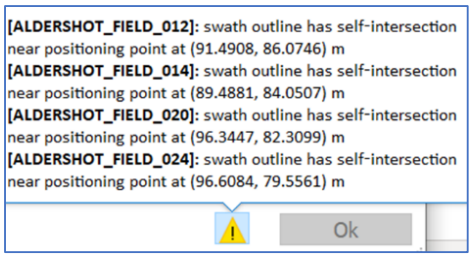

Figure 11, Additional statistics showing self-intersections in data

Figures 7 and 8 illustrate how this project will look after following the methodology described above, both as a whole project and a section close up using zoom.

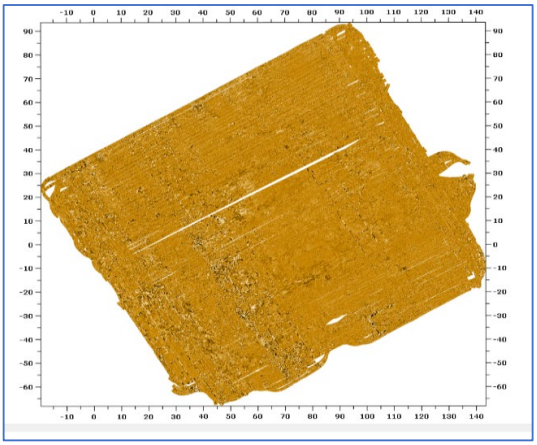

Figure 13, whole project, radar data shown

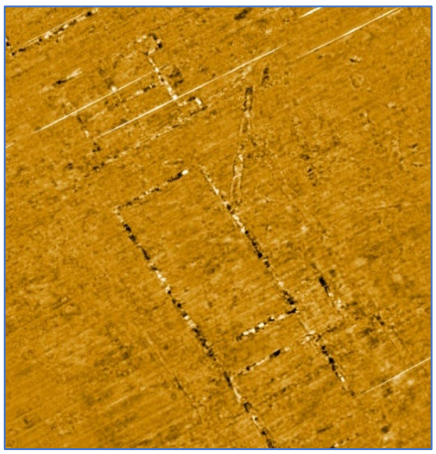

Figure 12, Zoomed section

An important takeaway from this exercise is that the geometry requires cleaning as much as possible before the import of radar data. Processing radar data takes up a lot of computer memory and can slow operations down. In this instance, none of the geometry was moved as that would be very difficult since we have no references, before radar data import. The next note will show how we may use visible object in the radar data to correct for some positioning errors, besides some other hints.

Interpolation and Positioning Correction of Raptor Data

In the previous note, we covered some steps for cleaning up geometry before loading radar data; the main idea was to save time by not going through the process of loading a large amount of compromised data. This note will deal with the possibilities to edit geometry after loading radar data and explain why, in general, positioning needs to be very good in 3D-projects.

Loading of radar data

When we load radar data, we must specify the interpolation distance the software will use internally. As mentioned before, it's usually of little use to make this distance shorter than half the channel spacing.

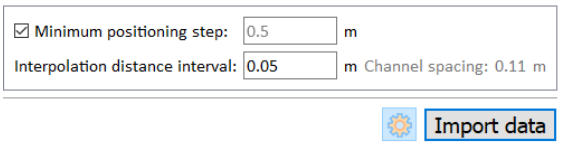

Figure 14, Data import parameters, reduction of positioning points and interpolation distance

Another factor is the memory needed for managing the data, and this distance directly dictates that. In Figure 3, we show a small project, with raw data of 55 MB. During data collection, the point distance was 2 cm, with a channel spacing of 11 cm. The images show data interpolated to 2 cm, 4 cm, and 10 cm bins. As can be seen, to locate the utilities, any one of those settings would be just fine. However, the disk space needed to accommodate all steps in the post-processing up to the stage shown is quite different – 0.25 GB for 10 cm binning and 3.8 GB for 2 cm binning. So, in this case, by interpolating to 2 cm, we ended up with 70 times the original data size. However, any visible benefit is negligible, so interpolation with 10 cm binning seems a suitable choice, given that it only requires 5 times the disk space for the raw data.

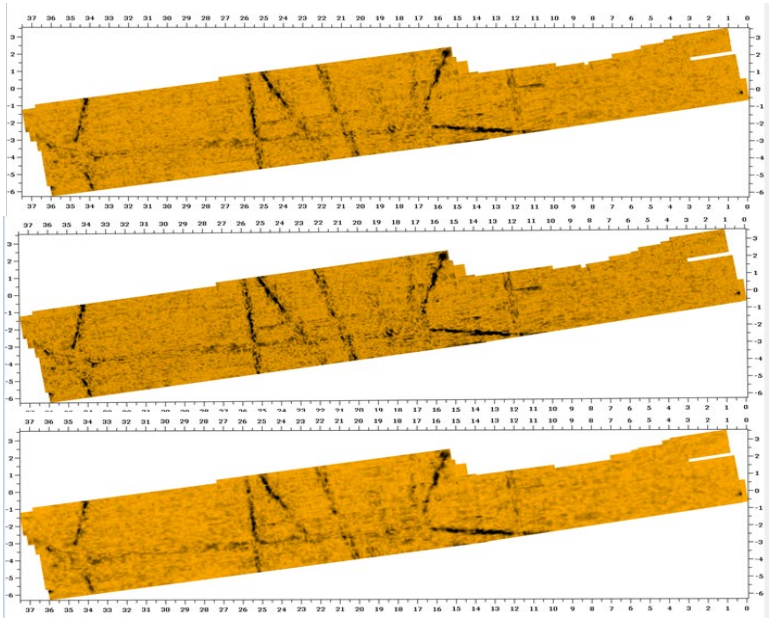

Figure 15, a small project, raw data takes up 55MB, interpolated to 2cm, 4cm and 10cm bins

Do we need some fancy filtering for importing the data? No, dc-adjustment, de-wow, or bandpass, combined with threshold and compensation for the Rx-Tx distance, is all that's needed – assuming, of course, the raw data is of good quality.

Correcting bad positions

Figure 3 below shows a section from a survey conducted with a vehicle-mounted array. Not even the most erratic driver could create the track A-B-C as shown. This type of positioning error is typical when

you allow for variations in RTK-coordinates between fix and float. When the fix is lost, the output from the GPS jumps unpredictably.

We don't know whether the positioning before B and after C is good, but we can conclude from the anomaly at A that we're not entirely lost. At A, we have a continuous anomaly crossing over two swaths, so at least the relative position between these two swaths is good at that point.

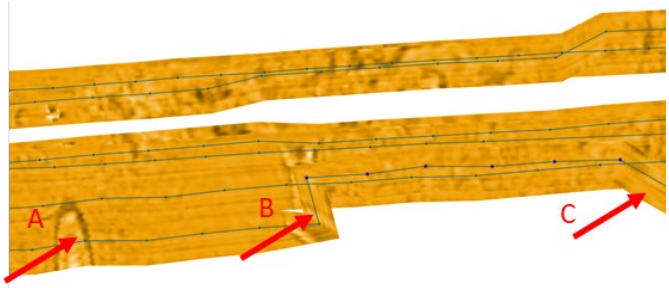

Figure 16, a section with clear positioning errors, marked by red arrows at B and C

In Figure 4, the result of the correcting actions is shown, with some higher gain on the data. We now have a continuous anomaly at B and can be sure that we did something in the right direction. What we did here was mainly to delete positioning points between B and C, leaving the odometer wheel as the only positioning device between these points.

So can we now conclude that it's possible to fix bad positioning? No, we should not think in that direction. It's possible to correct minor errors, but if the positioning is bad throughout a project, it will be too time-consuming to fix. Recall the project shown in a previous note, with more than 70,000 positioning points; it would be impossible to correct a large chunk of those.

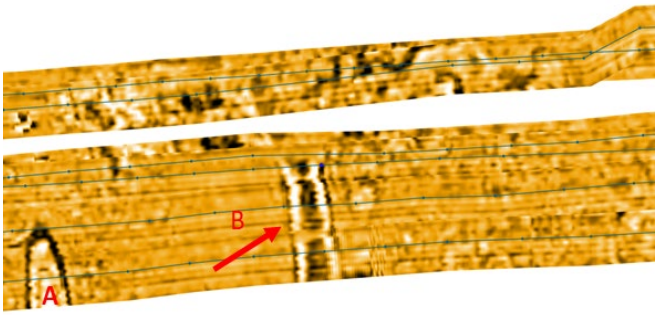

Figure 17, same section as in previous figure but with corrected positions and some higher gain. Note the now continuous anomaly at B

In practice, we're limited to deleting some visibly wrong positioning points and moving others, provided we have anomalies to aid in doing so. Having said all this, we should also mention that commercially it's often ok to live with some minor errors, and the ability to correct some of the geometry may not always be worth the effort.

Takeaway

Often interpolation distances are chosen too short in the belief that this will enhance the data, while in fact, the channel spacing is the most limiting parameter. We're not saying that one should always interpolate to the channel spacing, only that one should not overestimate the ability to use the seemingly higher density along the swath to enhance the final images. We haven't seen any significant benefit in interpolating to less than half the channel spacing. We've also noticed that a modern, interactive software makes it possible to correct for some positioning errors, although we also warn for over-optimistic views on this ability. In large datasets, it's impossible, and when it is possible, it relies heavily on having anomalies visible.

In our next note, we'll cover some of the processing we do before the interpretation and export stages.

Processing of Raptor 3D GPR Data

In the previous notes, we've covered a few steps concerning the cleaning up of geometry and the loading of raw GPR array data. In this note, we'll deal with some necessary processing steps before data interpretation.

Raw data

Figure 1 below shows the raw data from one section of a project. The lower image shows the data with an overlay of the geometry. This data set is by no means ideal. Firstly, there are quite some gaps in the data. Secondly, and more striking, is the inconsistent coverage, with different orientations, collection patterns, and overlapping data. Some filtering has been applied in the form of a dc-removal filter, and a threshold level for time-zero alignment. We'll see now how it looks after a few simple steps.

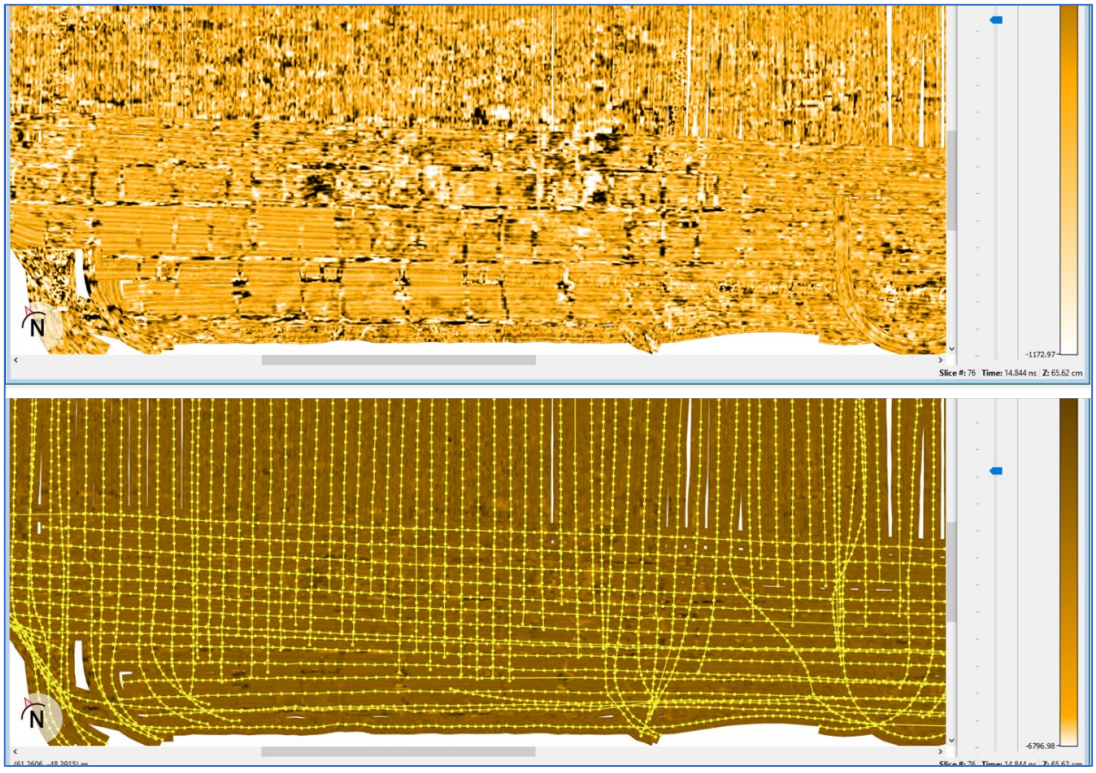

Figure 18, Raw data (top) with geometry overlay (bottom). Note the crisscrossing of lines and non-symmetrical coverage

Step 1 – pre-processing

Figure 2 shows the available pre-processing routines, of which the following three are the most commonly used.

- Antenna ring-down (500 traces in background removal)
- Bandpass (170 – 600 MHz)
- Amplitude correction (spherical divergence correction with no parameters)

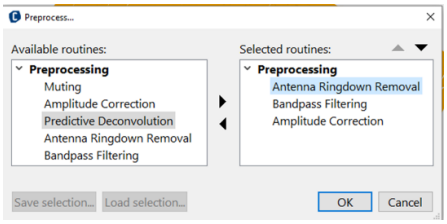

Figure 19, Pre-processing routines

The defaults for each routine are well defined, so there's usually no need to change settings, just select and run.

The effect of these basic pre-processing routines is shown below in Figure 3. In this case, one may think it's possible to start the interpretation here. However, we're only viewing a single depth slice now, and deeper ones may be much more blurred. Thus, we recommend continuing to the post-processing.

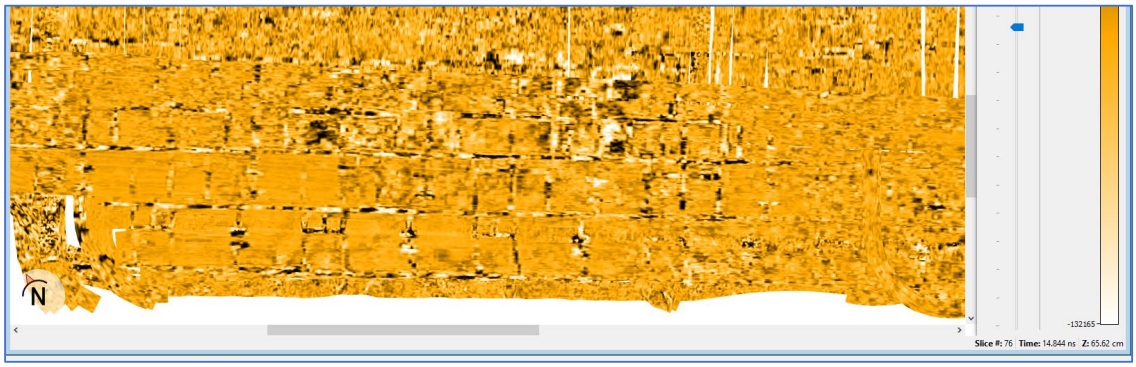

Figure 20, Top view after pre-processing with background removal, bandpass and amplitude correction

In the original data from Figure 1, linear (humanmade) features were visible, but the pre-processed data in Figure 3 represents a significant improvement. Striping in the data is gone, and most features are more apparent.

Step 2 – Regularization/ interpolation

To this point, only 1D and 2D routines have been used. To apply a 3D-migration routine, we need to interpolate the data into regular bins (the size of which (4 cm) was selected when loading the data). At this stage, gaps are also filled by interpolating data from adjacent points.

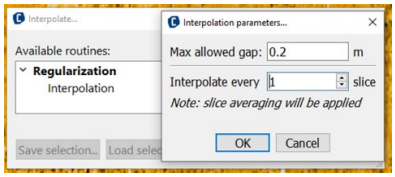

Figure 21, menu for selecting interpolation parameters

Figure 4 shows the options for the interpolation routine, including the maximum gap the software will attempt to fill. The rationale behind this parameter is that there's no use in trying to fill in significant gaps with any interpolation algorithm, as it won't work if the gap is too large. Then we also have the option of slice averaging, which may save some significant disk-space. Figure 5 shows the result after interpolation. *Note* – this stage is usually the most time-consuming routine applied to 3D-data.

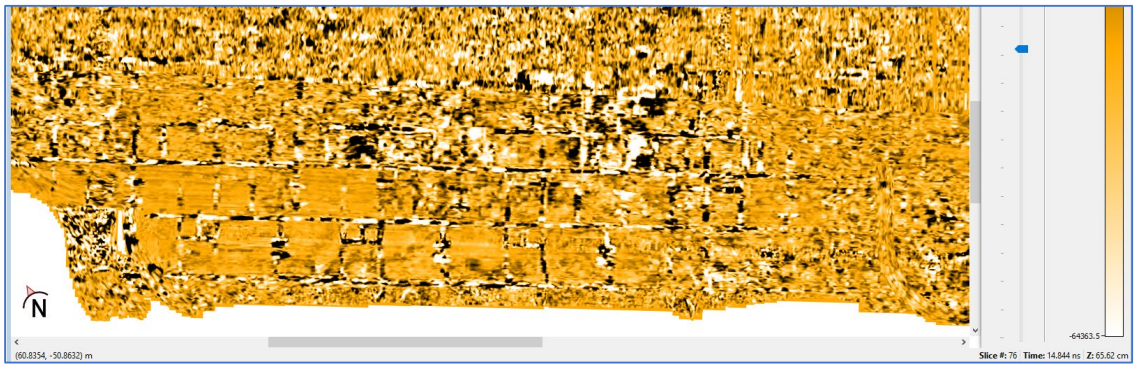

Figure 22, the result after the Regularization/interpolation stage

Migration and post-processing of GPR Data

Migration is the process during which hyperbolic anomalies are collapsed into points with the help of a known velocity and a selected algorithm. There are a few mathematical algorithms to choose from, each with their pros and cons. However, the key to dealing with 3D-GPR data is to apply a true 3D-migration after interpolation. The alternative is to use 2D-migration and then interpolate the migrated profiles into a 3D-volume, but this approach does not give the same excellent results.

When applying migration, we need to know to which velocity. Instead of guessing, we can use an interactive tool to select the optimal value, and that process will be discussed later in a separate technical note. For now, we jump directly to the post-processing stage.

Figure 6 shows the available post-processing routines, the majority of which will be discussed later in a separate technical note. For now, we concentrate on the commonly used Amplitude Envelope.

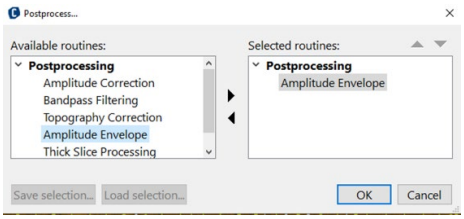

Figure 23, post-processing routines

Amplitude Envelope is the process of applying a Hilbert transform to the data. In simple terms, this effectively moves negative data values over to the positive side and draws lines between peaks. It may reduce resolution slightly, but this is generally acceptable due to the more straightforward interpretation that follows.

Of course, in modern software, a user can always jump between the different processing stages to make use of higher resolution available in other data instances, although this is rarely needed.

Figure 7 displays the impact of migration and the post-processing routines on the working data example. Following a few simple processing steps, we now have data that is much easier to interpret. Again, this is only one depth-slice, so scrolling through the entire depth range, will reveal all targets.

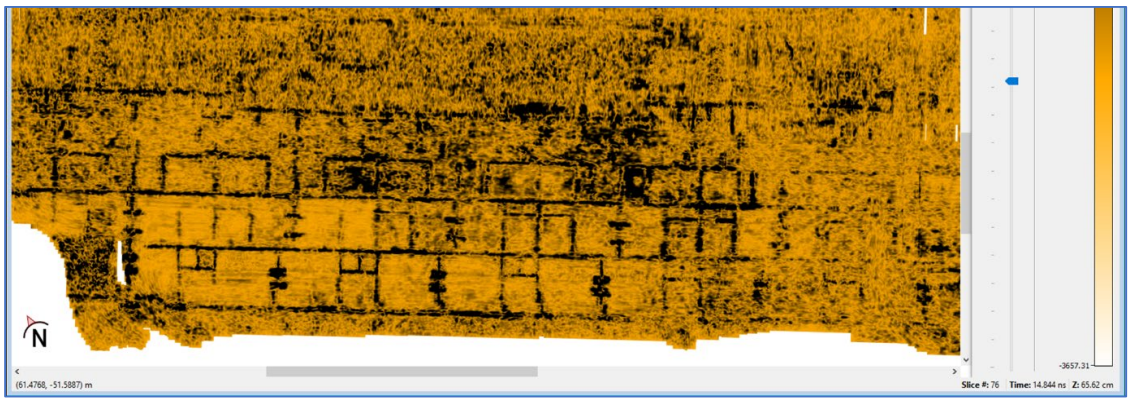

Figure 24, migrated, and post-processed data (at the same depth as in previous figures)

Takeaway

Modern and interactive processing software makes Raptor 3D GPR data easy to manage. You don't need to be a scientist; just follow a few simple guidelines, and the resulting data is significantly more straightforward to interpret than ordinary 2D GPR data, and the ambiguities are gone!

Efficient 3D-migration of Raptor Data

In the previous note we covered the processing steps up to post-processing, jumping the migration procedure. In this note we'll look specifically at migration and how modern and intuitive software makes this a straightforward process.

3D-migration

Migration is the process by which hyperbolic anomalies are collapsed into points using a known velocity and a mathematical algorithm. Nowadays, it's not used extensively in 2D-profiling, since operators learn quickly to recognise the hyperbolic shapes commonly formed by utility lines and other targets.

In our view, the value of migration is most pronounced in top views (C-scans) of data, where linear targets and edges spread out laterally in un-migrated data. The process also has the potential to clean up the asymptotes from the sides of ditches or other buried targets, which are not necessarily linear. It's worth noting that in the past it was quite common to migrate 2D profiles and then apply interpolation to the migrated sections, but this is not what we mean with true 3D processing.

To make a stringent migration, one must know the wave velocity throughout the whole surveyed site. In 2D-data, it's possible to apply a layered velocity model before migration, while in a large 3D-data set, this becomes very difficult, if not impossible. Add to this that migration is a quite time-consuming process and that we know of no software able to handle variable velocities over large areas effectively. So, we should have some tools and strategies to make this process smooth and effective.

It's common to apply hyperbola fitting on 2D sections to estimate local velocity. While this may work well in many cases, we propose a more robust method, via test migration of selected 2D sections.

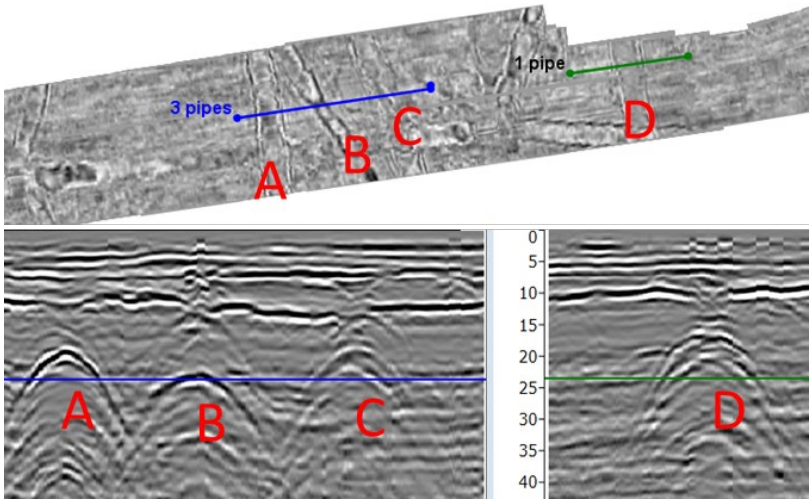

Figure 25, Top: C-scan with marked 2D cuts across four linear targets. Bottom 2D-views of the cuts marked in C-scan. Horizontal lines in the 2D views shows the depth/time of the C-scan

Modern and interactive software will allow the user to select and visualise any 2D cut from the top view. As illustrated in Figure 1, we see two lines crossing four linear targets and the corresponding radargrams showing the hyperbolas. While the first three targets may easily be estimated with hyperbolas, although a keen eye will note the variations in the diameters, the fourth does not look that clean at all. It may be the roof of a culvert rather than pipe.

Method

Since 2D migration is fast enough to make interactive tools possible, we provide a slider for varying the velocities in the radargrams shown in Figure 1. Then we fine-tune the velocity to find the setting that compresses the anomalies the most. We don't think about this as finding the true velocity in a stringent way, but rather to compress the hyperbolas the most. Figure 2 shows a result when focusing on targets A and D, where the resulting velocities are 75 m/µs and 84 m/µs, respectively. So, which one to use? A variation of almost 10 m/µs is quite significant, and a velocity of 75 m/µs is considered low, in a case like this. Now, it's easy to find out by using the slider to adjust the velocity up to 84 m/ µs, which gives the result shown in Figure 3. Now target A shows the typical 'smiles' characteristic of too high a velocity, while the other targets compress better.

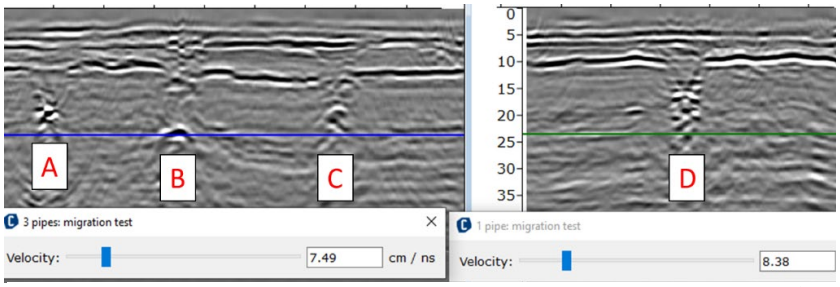

Figure 27, Test migration of the two radargrams from figure 1, left migrated with 75 m/µs and right with 84 m/µs

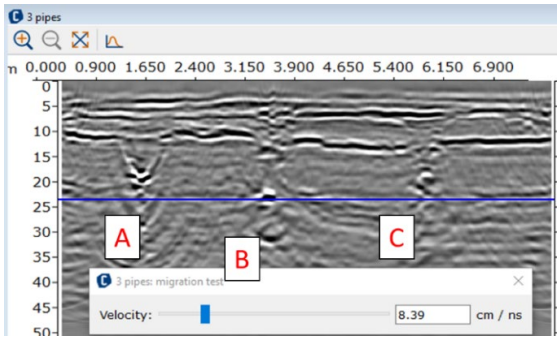

Figure 26, first radargram migrated with velocity set to 84 m/µs shows better compression of target A and B, but starts to look over-migrated

It's possible to delve into a lot of detail concerning the correct velocity adjustment. For example, the profile doesn't cross each target perpendicularly. However, there's often a compromise to be made, which in reality is not crucial, as long as migration velocity is treated separately to the velocity used for depth calculations and awareness of the compromise is kept.

Figure 4 shows the result of 3D-migration with velocity set to 80 m/µs and the targets compress nicely, regardless of their directions. The latter is the strength of true-3D migration based on proper channel spacing, good positioning, and practical software able to stitch/bin the data into a 3D volume.

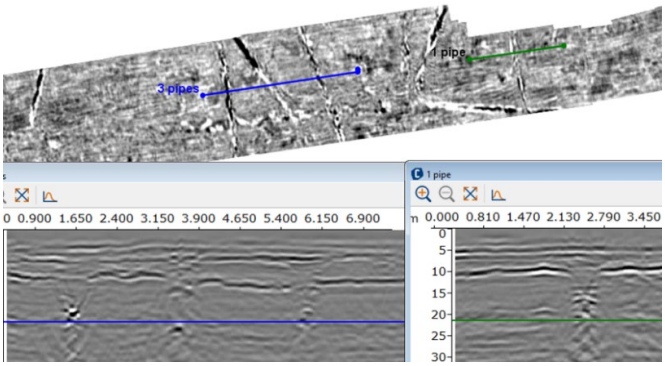

Figure 28, result of 3D-migration with velocity set to 80 m/µs

Takeaway

In 3D-applications, the migration procedure shines when it comes to visualising targets in the top-view because when working with interpretation, it's the view most utilised. There's no deep expertise needed for applying it correctly when modern, interactive software makes the process swift and intuitive. In this example, we did not

show the full strength of it, and that will be more obvious when we come to AVI-exports and deep-slice processing, and those discussions will follow in subsequent notes.

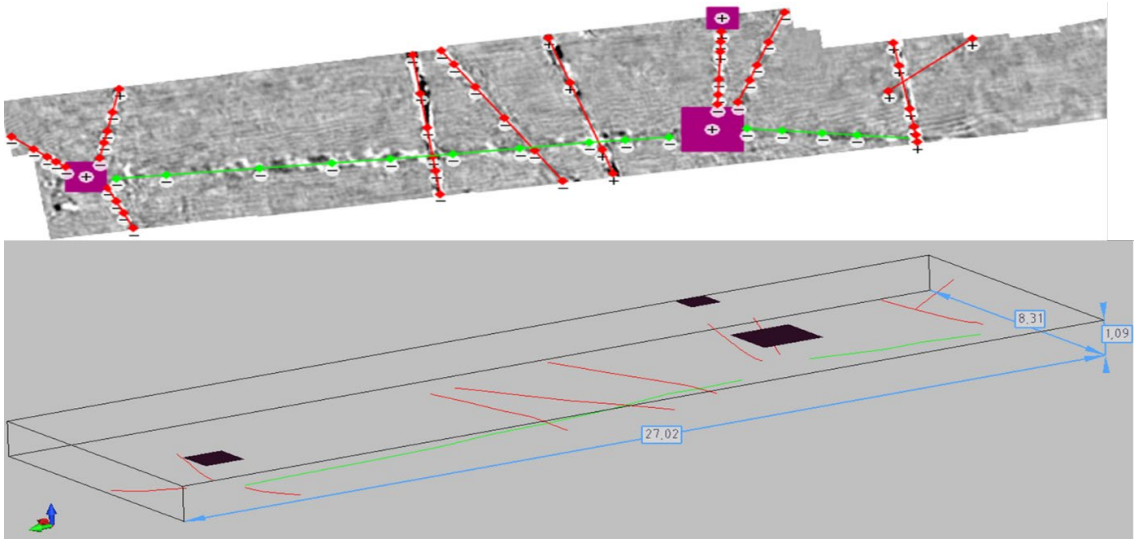

Figure 29, Final interpretation as it might look in the radar software, top. Exports to a dxf-viewer, bottom.

Interpretation of Raptor Data

In the previous note we briefly covered data collection and the most common processing steps. We concluded that if the raw data is of good quality, especially when it comes to positioning, the following processing can be swiftly done with modern software. That brings us to the most time-consuming step in managing 3D GPR data – interpretation. This stage is a real bottleneck and where good software can make a significant difference. In this note, we look at the simplest, but a reliable, way of interpreting Raptor data.

Top-views are probably the most common when it comes to interpreting 3D GPR data. However, they are not that useful for the precise picking of target depths. Instead, their strength lies in giving the user an overview and the perception of the target layouts. Having 3D data at hand provides us with the ability to view any 2D cut in that data volume. If those cuts are made properly, then picking a target in the 2D view, combined with views and picks in the top-view, makes the process more accurate.

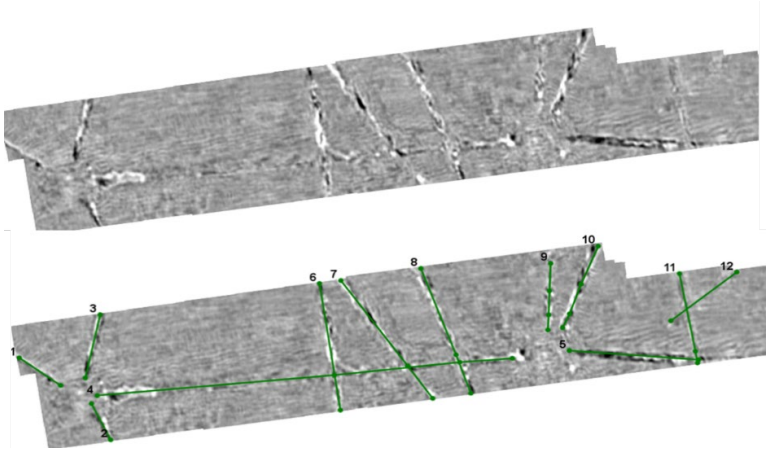

Figure 30, Clean top-view at depth slice showing most targets and laid out cut-lines(green), bottom

Figure 1 shows a top view where many targets are visible at the same depth slice. The cut-lines suitable for target picking are shown below. Laying out these cut-lines is intuitive, and the ability to scroll up and down in the time-slices makes it straightforward to place them correctly, centred on the targets. Once in place, it is possible to pick a straight utility line in a 2D view in a matter of seconds. Curved and dipping targets will be a little more time consuming to pull out.

During this process, the software must support an effective workflow, because, in a complex project, the screen can quickly become cluttered and confusing to understand. Things which may not seem significant, when working a small project, now reveal their importance. For example, having clear positioning indicators, minimizing keyboard inputs, auto-naming, auto-colouring, short-cuts, the ability to switch between different processing instances and views easily, a simple tool for measuring distances, and a straightforward means to turn such tools on and off, are but a few to mention.

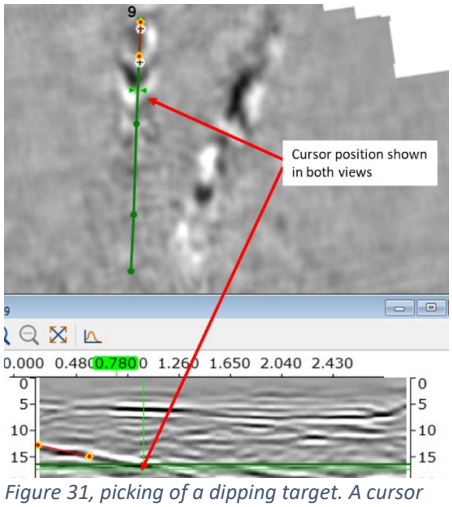

Figure 31, picking of a dipping target. A cursor shows the actual position of the cursor along the cut line and a horizontal line in the 2D-view show the actual depth slice.

In Figure 2, the picking of a dipping target is shown. The horizontal line in the 2D view keeps the user aware of where the depth slice is located in the top view, with the cursor positions shown in both aspects. Needless to say, modern software must allow interpretation in all the available views, without restrictions.

In Figure 3, a slightly more complex situation is shown. Here we are marking a target which crosses under another line. In cases like this, and even more complex ones, the software must give the user practical tools for navigating through the data to manage the views and interpretation features.

A user might want to add manholes or other infrastructure visible in the data, assuming they did not bring them into the project as surface features during data collection. This ability can add value for the final touch up, likely done in a CAD environment; it may also be a useful QA/QC of the results. A final interpretation may look like the upper part of Figure 4, where for clarity, we also show a dxf-export with a bounding box.

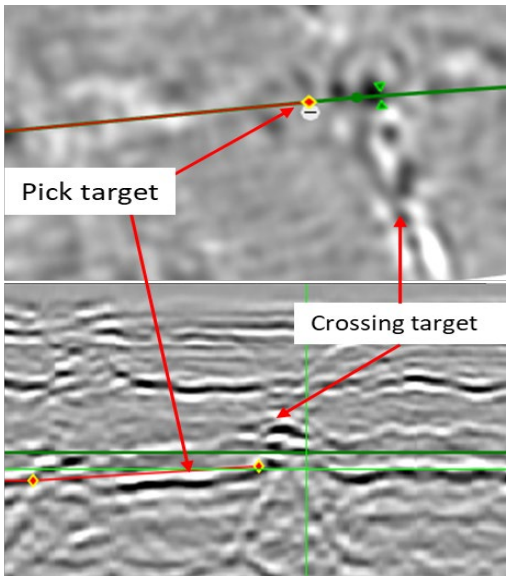

Figure 32,, picking a target along the horizontal cut-line in top view while crossing under a target coming in from 90 dearees.

Takeaway

The combination of 3D GPR array data and modern software removes many of the ambiguities often faced by users of simpler 2D systems. The dense data makes it possible to view the subsurface from any direction and thereby secure a reliable interpretation. Nevertheless, in larger projects, it is probably the most time-consuming part of the whole mapping process, which makes the user-friendliness and workflow support of a modern software critical.

We have shown here the most straightforward approach and left out more advanced tools and methods, which we will cover in the next part of this note**.**

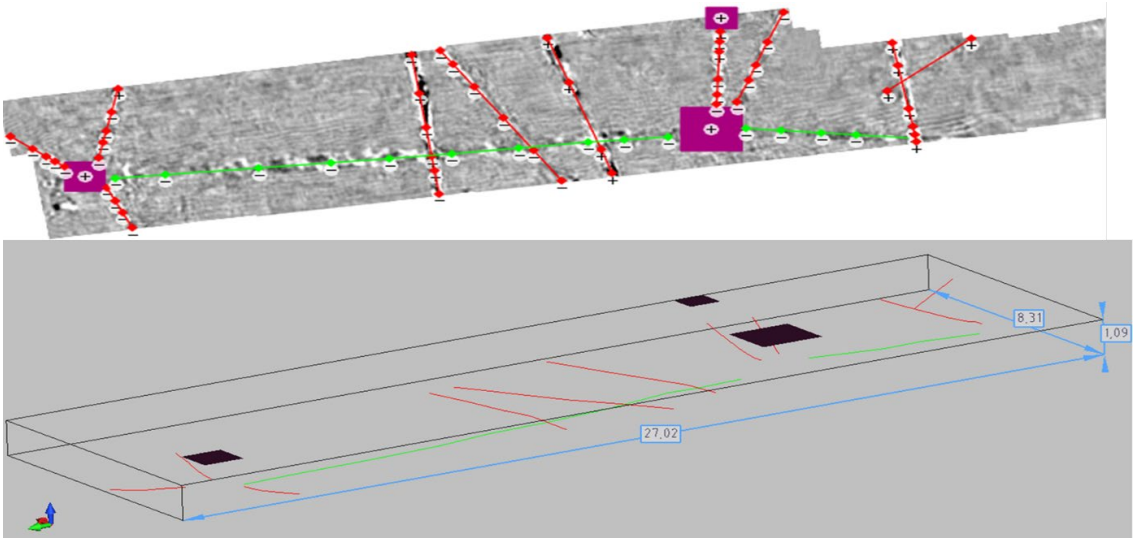

Figure 33, Final interpretation as it might look in the radar software, top. Exports to a dxf-viewer, bottom.

Interpretation of Raptor data - part 2

This note will delve further into 3D-GPR data interpretation and touch on the theme of quality control of the resulting exports.

Condor's development goal was to create 3D GPR processing software that was much easier to use while still being effective. When considering the different elements of a 3D-GPR project, e.g., data volume, target complexity, positioning systems, varying soils, applications, etc. etc., keeping the processing software efficient and user-friendly seems conflicting and complicated in itself. However, we have a growing and experienced user-base, helping us refine this process to suit typical workflows.

On that note, we introduce a feature called 'ribbon-box', which allows the user to define a volume on the top-view with a few simple clicks (as with the 'cut-lines' discussed previously). Figure 1 below, demonstrates the concept; where the ribbon-box's outline, shown in red, is defined when the user clicks the centre line shown. In this case, we define the box with nine vertexes, and the

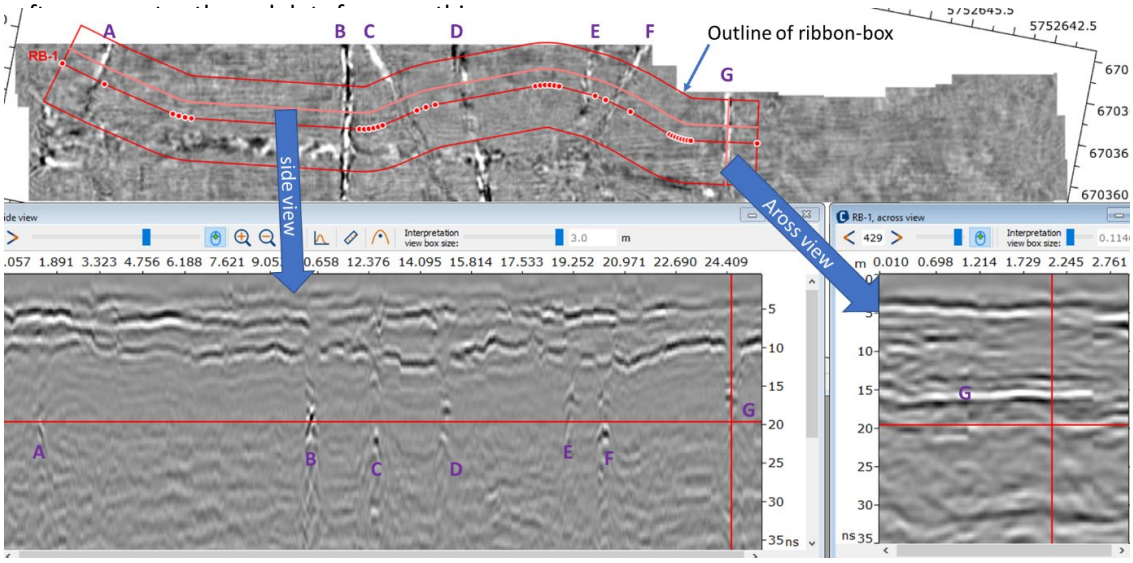

Figure 1, Top, layout of ribbon-box with marked utilities, bottom; side-and across views with the same utilities labled

Once the last vertex of the box is clicked, the two bottom views appear, showing a side-view and an across-view. The side- view can be moved freely in the box but is always parallel to the centre line, and the same is true for the across-view, which is perpendicular to the centre line. Both cuts are shown in a brighter colour in the top-view.

In Figure 1, the side-view shows the marking of seven utilities, A to G, which are also clearly visible in the top-view. The outline of the ribbon-box is created so that it crosses the marked utilities at approximately 90 degrees. This placement means that the across-view will be parallel to the utilities when moved to intersect them, an advantage when working with precise target picking, shown below.

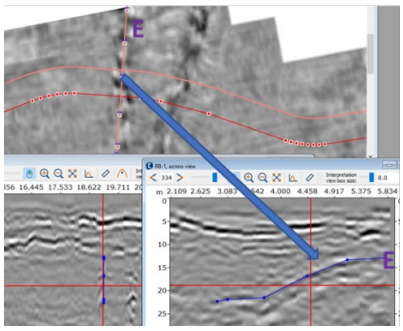

Figure 2, Target E highlighted by moving the across view to its location.

In figure 1, the across-view is right on top of target G, which is shown as a straight non-dipping anomaly, and easy to pick. However, scrolling the across-view over target E shows it

dipping strongly, as shown in Figure 2. It is now equally easy to pick this target in the across-view while checking the alignment and/or adjusting it in the other views.

By selecting a suitable outline and width of the ribbon-box, one can move through the data set and conveniently pick all the targets observed. This approach is faster and more precise than the use of simple cut-lines, as previously described. Regardless, cut-lines have a place and can still be highly useful.

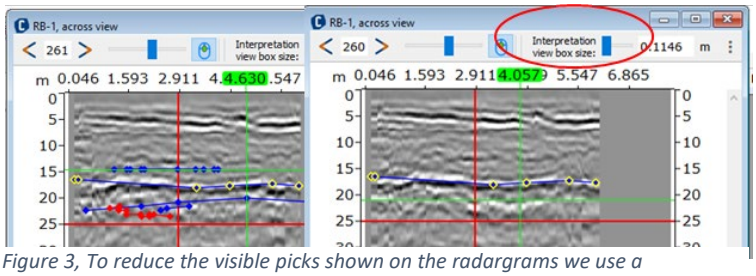

Figure 3, To reduce the visible picks shown on the radargrams we use a defined/variable volume, 'Interpretation view box size', left: large window, right: narrowed to the active pick.

A common problem when interpreting 3D-GPR data is that the number of targets can clutter the view, making it hard to know on which one you are working. To reduce clutter, you can toggle features on or off, but that means many repetitive mouse-clicks. However, when using the ribbon-box, you can adjust the depth of the interpretation view to mask picks outside of the working area, which easily declutters the data view, as shown in Figure 3.

Consequently, the ribbon-box tool is convenient and straightforward to use. Furthermore, it simplifies managing large data volumes, since it puts far fewer demands on the processing PC than a 3D-cube approach.

Once the interpretation process is complete, you can export the interpretations for further use to create deliverable reports. The most common format is DXF, as these files are compatible with CAD software typically used for this purpose. However, how these files where made, is something a user may have to explain to the end client. Of particular interest is the velocity used for calculating the depth to marked targets, since this crucial parameter determines the depth of interpretations.

As previously presented, Condor is constant velocity software that manages only one velocity at a time. However, it's essential to understand Condor's characteristics, which allow it to handle any number of velocities at different times, as shown in Figure 4.

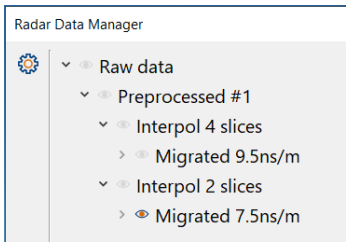

Figure 4, Condor manage different velocities, in this example it would be 3, if the operator does not intervene

Understanding the implementation:

- A project velocity is defined in preferences, and if left unchanged is the value used for all data instances up to the first instance of migrated data.
- Each instance of migrated data and any subordinate-instances are assigned the same velocity to which it was migrated.
- Every picked target is assigned the velocity used during picking, and that velocity is used when the depth of that target is written to the dxf-file.

Moreover, a log file for every picked feature, including the vertexes and velocity used while exporting it, may also be saved. The operator always has control and can overrule any pre-set velocity.

Ultimately, data interpretation depends on the user's skill and experience, but Condor dramatically aids the operator to make this process easier.

With access to the right tools, processing and interpreting 3D-GPR data does not have to be complicated nor tedious. With the advancement of modern 3D-GPR solutions such as Raptor, this technology becomes more accessible and therefore, the demands on interpretation tools increase. You should not need to be a geophysicist to do 3D-GPR work, and with the combination of Raptor and Condor, you need not.

Interpretation of Raptor data, part 3, Thick-slice processing/OspreyView

Having covered several useful visualization and interpretation tools in previous notes, we have not yet talked about 'thick-slice' processing. Therefore, in this note, we present a novel method for visualizing 3D-GPR data more effectively by applying 'thick-slice' processing. To the best of our knowledge, this method has not been commercially available until now. The original concept was presented to us by Mark Grasmueck et al., of the University of Miami. While Mark remains the innovator, we have adapted it as a feature within Condor.

Our efforts are to continually seek effective methods for managing huge datasets, without loss of resolution/detail, which preserve depth/position awareness and accuracy and, finally, lend themselves for precise picking of targets and exports to cad-environments. Further, our clients ask for such functionality packaged in a user-friendly environment, without the need for massive parameter tweaking. OspreyView, as we call it, meets all these conditions.

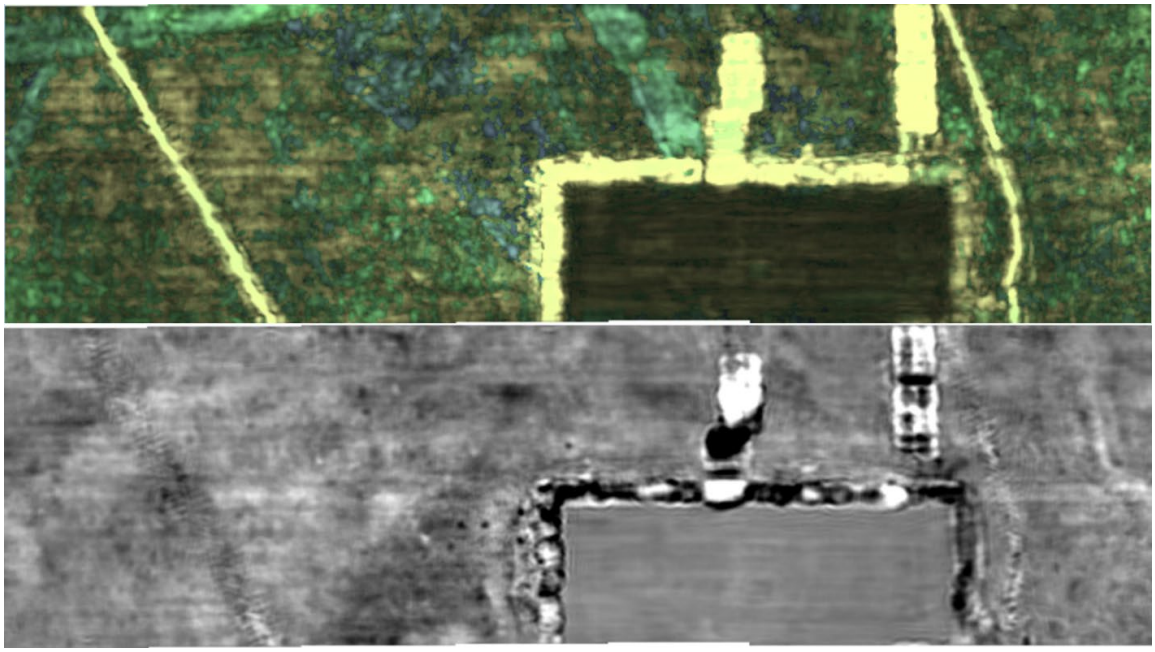

Figure 1, example of what OspreVIew brings to the table, top OspreyView, bottom traditional depth-slice. A significant improved overview, without resolution loss is obvious, and with encoding supporting a sense of depth. Raptor-45 data from a survey at a gas-station.

A first example of what we're presenting now is shown in figure 1. The top image shows our approach to 'thick slice' processing while the bottom view shows a traditional, thin-slice. Although both images are good, at least five utilities are shown clearly in the OspreyView image which are not present in the bottom one or only with some imagination.

Before delving further into details, let us briefly review some concepts in thick-slice processing. Firstly, a depth slice is a horizontal cut through your dataset, showing the data's amplitudes after the processing is applied. The resulting image (C-scan) has some significant characteristics. It gives an excellent overview which is very cumbersome to achieve through single line profiling, and it preserves the full resolution of the GPR-system and data collection. The depth localization of targets will be precise, and moving through the data will be swift. Figure 2 (top image) shows an example of such basic visualization.

As good as this type of 'thin-slice' visualization is, the obvious drawback is that it only shows what the GPR detects at the time/depth cut through by a given slice. There is no effective visualization of dipping targets, nor can it be possible to see those at entirely different depths. Scrolling is needed, which adds time, but more importantly, it does not give the operator a complete overview.

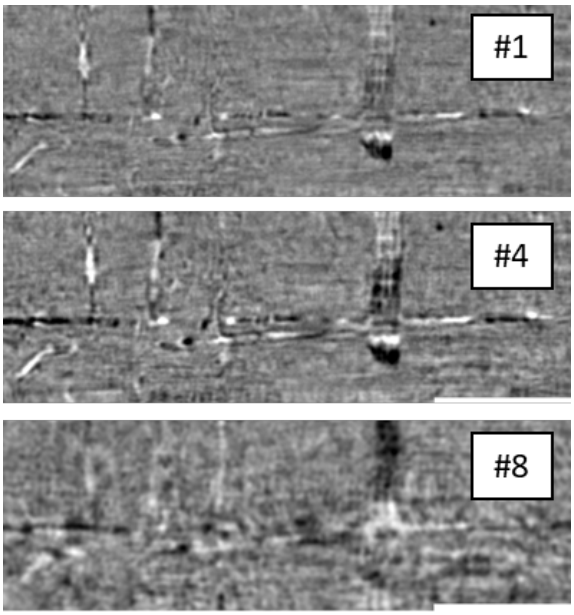

Figure 2, Top view on 1.3m, effect of slice-averaging, top no averaging, middle; 4 slices averaged, bottom 8 slices averaged. Raptor-45 data.

To overcome this drawback and get a better overview, it's common practice to add more processing by averaging slices over a specific depth range. In Figure 2, the middle and bottom images show the effect of such averaging over 4 and 8 slices, respectively. While the averaging applied to the middle image works, it doesn't work on the bottom image. This data's severe degradation is due to the averaging spanning, close to, a full wavelength of the radar signal. This method always reduces resolution, and even if the 4-slice averaging seems to work, a careful look will reveal loss of information. A Hilbert transform is often added before averaging to avoid the cancellation due to the GPR-signal's bi-polar attribute. However, this approach further reduces resolution and therefore isn't what we want.

That said, animations of moderately averaged slices work well. Set up at a suitable frame speed, they provide an effective way of visualizing all present targets, and this approach will likely not go away soon. Although the full resolution is preserved, it does not easily lend itself as a tool supporting precise target picking for export to CAD/GIS environments – a critical task for our clients. Regardless, it remains a meaningful way to visualize 3D data and to bring attention to what can be achieved with GPR.

We could go on to discuss amplitude thresholding or other means of simplifying and improving visualization, but that is beyond the scope of this document. The objective is 'thick-slice' processing, so, let's jump to OspreyView. Figure 3 below shows a second comparison between this new method (bottom) and a typical 'thin-slice' (top). The two data examples reveal utility lines beneath a busy road intersection, centred at approx. 1.3 m depth.

In this example, the operator set OspreyView to see through from 0.2 to 1.9 m, revealing almost everything available in this data. This extensive range may blur some subtle details, but it gives an excellent overview.

Given the example in Figure 3, the advantage of OspreyView should be obvious to anyone working with 3D-GPR data, even though only the birdseye view through the ground is shown. This birdseye view is not the final delivery our clients require, but having a clear overview of the scene is essential. We will talk about precision later.

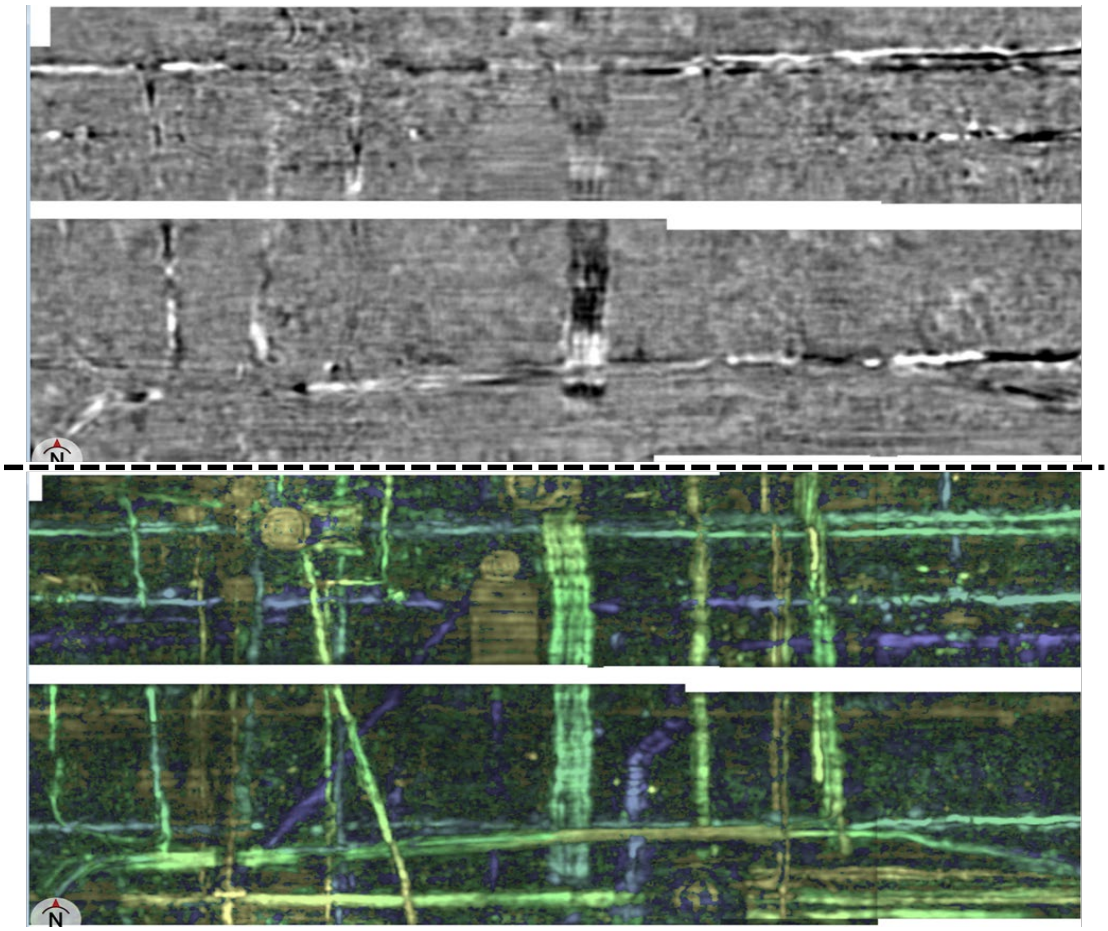

Figure 3, Comparison between slice averaging by 4 (top) and OspreyView (bottom), at 1.3 m depth. Yellow means more shallow, bluish deeper and green centred. Raptor-45 data.

How does OspreyView work?

In simple terms, the software applies a colour matrix for decoding depths and signal return strengths. This approach is different from the 1-dimensional colour palettes commonly used throughout the GPR industry.

As shown in Figure 4, the default colour matrix is set, so that targets at the centre are greenish, while those shallower or deeper are yellowish or bluish, respectively. This scheme is suitable for preserving a feeling of depth in the images (as per the inventor's concept). It's a straightforward process to alter the colour matrix, as needed for those wanting something else.

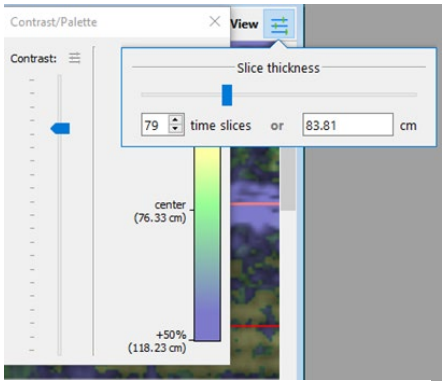

Figure 4, OspreyVeiw is controlled by 2 sliders, depth range and contrast.

User-friendliness is an often-overlooked feature. We think it is essential and strive to make that a hallmark of the ImpulseRadar brand – why make things complicated when they do not need to be? OspreyView is controlled by two sliders, one for the depth range, and one for contrast, as shown in Figure 4, above. The view itself is activated via a checkbox, so switching between the regular view and OspreyView is quick and couldn't be simpler.

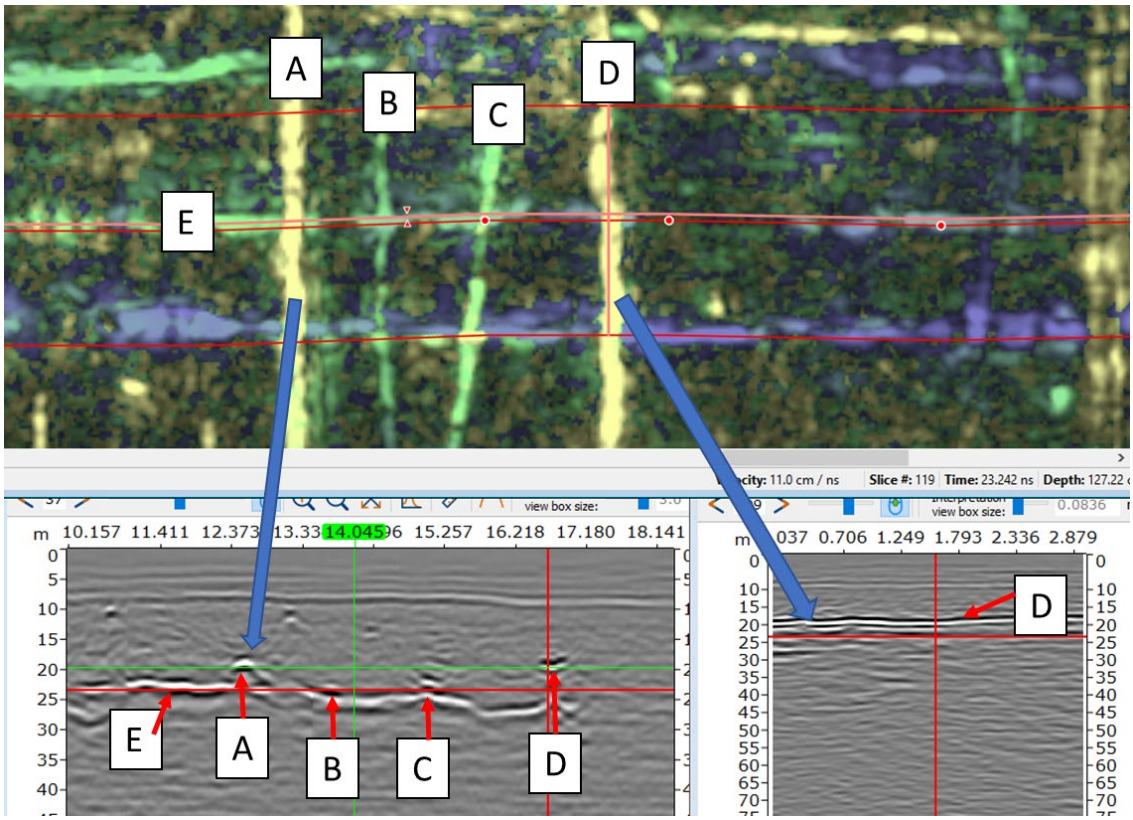

Figure 5, OspreyView combined with the ribbon-box. Five targets are marked in the data, all showing up on the side-view and one in the across view. Two of the targets, A and D, are in a different depth from the others, but still clearly visible.. Raptor-45 data.

Going back to what our clients need to put in their delivery reports, i.e. precise target locations, Figure 5 above, shows how OspreyView helps extract just that. The precision interpretations revolve around the Ribbon-Box function presented in earlier notes, and OspreyView blends seamlessly with this function. The result is that precise picking of targets is now an even easier task given OspreyView's superior colour-coded depth profile overview.

The OspreyView is not only a way to colour the top view, it also holds depth information. This means that when we pick a line along a coloured utility in the top view, the vertexes of those picks will be placed at the precise depth determined by that colour. So, in short, we now make precise depth pickings from the top view! For this to work, we must have reasonable clean data, on the other hand, any misaligned picks can easily be adjusted in a side-view.

We have also found that this way of visualizing data helps us find subtle targets and anomalies otherwise challenging to detect. They are not invisible in the single-slice view, but they are easily passed by unnoticed, if only visible at one depth-slice. Figure 6, below, illustrates this perfectly. The top image is an ordinary slice where some of the human-made structures are barely visible. They are only possible to detect in one or two slices, and weakly. Contrast this with OsrpreyView where they are clearly visible.

The usefulness is not limited to ideal data, and in fact, we have found it useful in all the projects to which it has been applied. Obviously, it does not help when a position is erroneous or when the soil is rendering GPR useless, but we have found it helpful even when the going gets tough.

Summary

OspreyView is the first commercial application of a novel method of visualizing 3D-GPR data. It is available in our CONDOR software, and the advantages are summarized as follows:

- The user has a clear overview of the targets beneath, instantly, without further processing in an arbitrarily variable time window.
- Completely preserves the resolution of the original data, both in depth and spatially.
- Very fast, flipping between traditional slice-view and OspreyView is instant.
- Supports precise picking of targets, including correct depth, from the top view only, while not interfering with the pickings in other views.
- Does not require separate processing instances, no extra disk space needed.
- Helps in detecting faint objects, minimizes the risk of missed targets.
- Has a straightforward and intuitive user interface, no parameter-tweaking needed.

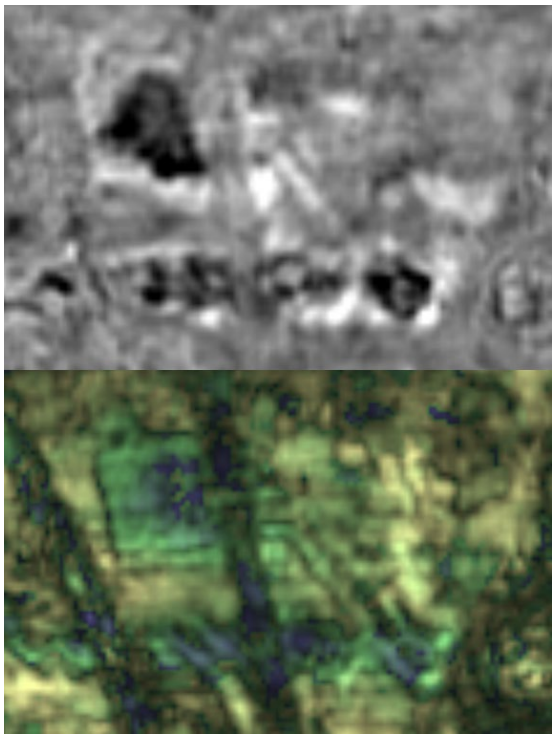

Figure 5, Example showing how OspreyView helps in revealing subtle details in data, 0.5m depth, Raptor-45 data.

impulseRADAR

ImpulseRadar defines GPR

www.impulseradargpr.com