Augmented Geophysical Data Interpretation Through Automated Velocity Picking in Semblance Velocity Images

J. Ross Beveridge, Charlie Ross, Darrell Whitley
Computer Science Department
Colorado State University
Fort Collins, CO 80523
ross@cs.colostate.edu

Barry Fish
Sun Microsystems
(Previously at Landmark Graphics)
Denver, CO
Barry.Fish@central.sun.com

Abstract

Velocity Picking is the problem of picking velocity-time pairs based on a coherence metric between multiple seismic signals. Coherence as a function of velocity and time can be expressed as a 2-D color Semblance Velocity *image. Currently, humans pick velocities by looking at the Semblance Velocity image; this process can take days or even weeks to complete for a seismic survey. The problem can be posed as a geometric feature matching problem. A feature extraction algorithm can recognize islands (peaks) of maximal semblance in the Semblance Velocity image: a heuristic combinatorial matching process can then be used to find a subset of peaks which maximizes the coherence metric. The peaks define a polyline through the image, and coherence is measured in terms of the summed velocity under the polyline and the smoothness of the polyline. Our best algorithm includes a constraint favoring solutions near the median solution for the local area under consideration. Each image is first processed independently. Then, a second pass of optimization includes proximity to the median as an additional optimization criterion. Our results are similar to those produced by human experts.*

1. Introduction

Velocity picking is one of many steps used to construct maps of the earth's crust from acoustic data collected on the surface. Currently, geophysicists spend hours, days or weeks in front of monitors tracing curves on images. Velocity picking involves placing a polyline through a series of high energy locations subject to global constraints on the shape and placement of the polyline. This would seem an ideal task for a computer vision algorithm.

Our algorithm starts by identifying peaks in a semblance velocity image. Usually, some subset of these correspond to the bright spots selected by the geophysicist. Next, a combinatorial optimization algorithm searches for a particular subset of these. For each subset, there is an associated curve, where the curve is represented by a polyline formed by connecting peaks starting at the top of the image and moving down. There is a combinatorial explosion in the number of possible polylines, so a heuristic search algorithm is used to find one that is near op-

timal as measured by a criterion function we define. The criterion function being optimized captures some of the factors taken into account by the geophysicist.

Our solution to velocity picking is novel in several respects. One is the problem-specific representation for the curve and the associated local search neighborhood used by our optimization process. Another is our finding that a solution guided by the median of many inexpensive solutions is superior to any other method tried. In particular, it was superior to taking the same amount of time and allocating it to a single more thorough search.

Solutions developed by our automated algorithm have been compared to human generated solutions on 26 semblance velocity images. The automated solutions are not identical to those generated by humans, but they are sufficiently close to be apparently equivalent to a trained eye. In addition, our algorithm is interactive and the human expert can easily provide additional guidance when needed. We are confident this tool is now ready for use in industry.

2 Background

During a seismic survey, a source (e.g. a dynamite shot) and multiple receivers (e.g. geophones) are used to record seismic signals. Data from many sources and receivers arrayed over the survey area provide indirect information regarding each "reflecting layer" of earth strata. To reconstruct the underlying strata, geophysicists must correct for the distance between different sources and receivers and combine data from multiple signals into a **common midpoint gather**. In effect, the common midpoint gather is a restructured signal that models what a seismic signal would look like if it had been collected by a source and receiver both sitting at exactly the same spot on the earth's surface.

To assemble a common midpoint gather, the average signal velocity of sound from the earth's surface down to a specific reflecting layer must be estimated. Guidance for picking velocities is obtained from a 2-D Semblance Velocity image which encodes the "power", or cross-correlation, between all signals involved in the common midpoint gather. For our purposes here, time encodes the depth to a particular reflecting layer. The greater the power for a velocity-time pair, the more coherent are the signals in the seismic survey.

The image on the left of Figure 1 is a Semblance Velocity image. The x-axis is velocity with velocity increasing to the right. The y-axis is time. For our purposes, time can be thought of as distance or depth. Depth is increases going from the top to the bottom of the image. The power associated with a particular velocity-time pair is usually represented by color: blue-green-yellow-red shifts correspond to a increasing power (blue = low, red = high). The

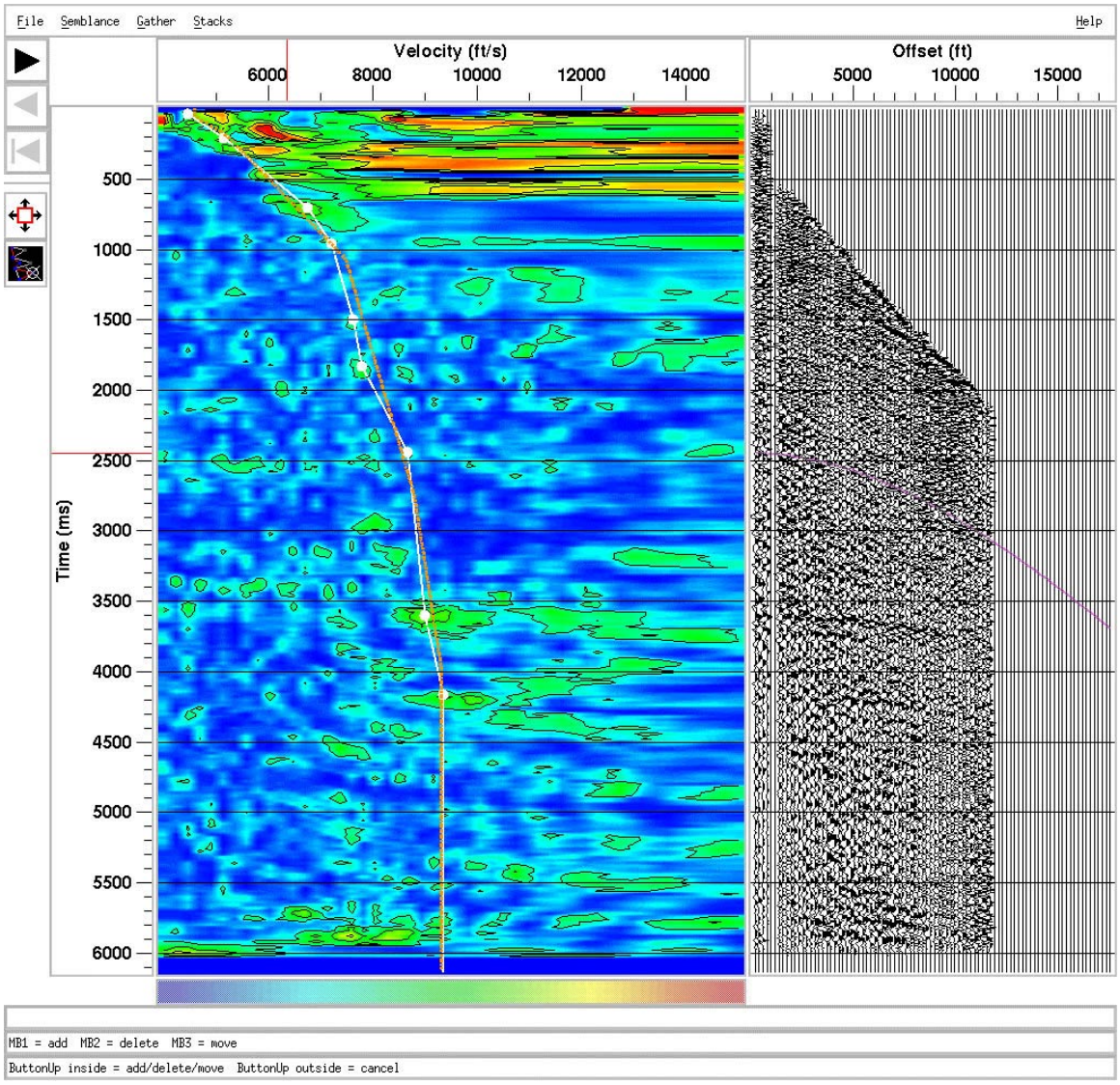

Figure 1. *Example of a Semblance Velocity Image with a human pick overlaid on top.*

high energy “islands” are the most likely velocity-time pairs corresponding to the geology.

While the “islands” representing peak power in the Semblance Velocity image suggest good velocity estimates, simply chaining together all plausible peaks is inadequate. There are many complicating factors, including echoes and artifacts produced from complex geophysical structure. Velocities usually increase with depth due to compression of the rock and selected peaks must satisfy certain structural properties in the Velocity image. They must also yield a high quality common midpoint gather, which is related to the quality of the adjusted signals shown on the right side of Figure 1. These signals on the right represent the acoustic signals themselves aligned by the currently selected semblance velocity curve. For example, the purple line indicates where all of the aligned signals suggest a strong echo: evidence of a transition in the underlying material in the earth’s crust at this depth.

Velocity Picking is largely considered to be an unsolved problem, with the state-of-the-art involving humans picking velocities based on the Semblance Velocity image and supplemental graphics (the right hand side of Figure 1) showing how selected velocities impact the collection of associated seismic signals. Picking velocities by hand for an entire survey can take days or even weeks. Landmark Graphics, the company with whom we partnered while doing this work, has developed neural networks for automated velocity picking [6], but their approach requires hand processing to generate the training data and the results are less than adequate. At least one large oil company has been using spline curves to fit a surface through the set of Semblance Velocity images representing a survey. This approach does not pick individual velocities, but rather creates a curve through each Semblance Velocity image. Since the spline approach abandons the connected peaks representation used by human experts, it is difficult to allow the human expert to interact with an automated velocity picking algorithm based upon splines.

In our approach, a set of initial candidate peaks are found in the Semblance Velocity image, and then a heuristic optimization procedure selects the best subset of candidate peaks. The polyline running down the image and passing through these peaks is the velocity pick. Identifying candidate velocity peaks in the Semblance Velocity image is an example of an image feature extraction task similar to peak detection as commonly performed in thermal imagery for Automatic Target Recognition [5] and closely related to model-driven feature detection [9, 7, 10]. The combinatorial search algorithm is a variant on local search which is highly adapted to this particular task [12].

3. Automated Peak Detection

Think of the semblance velocity image as a surface with each pixel representing the elevation at that point on the surface. Bright areas of the image correspond to mounds or hills and a peak is any pixel higher than its surrounding eight neighbors. From any pixel which is not a peak, there must exist a continuous energy path of adjacent pixels which monotonically leads to a peak. A neighborhood of size eight is used: North, South, East, West, Northeast, Northwest, Southeast and Southwest. This means that every pixel is in the energy basin of some peak. (Imagine every pixel sends out a scout who climbs uphill until it reaches a peak.) For our purposes, we define the *basin size* of a peak to be the number of pixels that lie in its basin of attraction. An efficient message passing scheme is used to compute peak basin size [13].

When two or more adjacent pixels have equal value and are otherwise local maxima, the pixel coordinates of all such pixels are averaged to obtain the position of the peak. When hill climbing is presented with two or more equally good choices, the first in row major order is taken.

Generally, more peaks are found than are necessary–especially very small peaks that are not significant features in the image. To reduce the number of peaks to a manageable number, the peaks are sorted according to a simple salience measure. Salience is the product of the peak basin size times peak height, where peak height is the semblance velocity at the peak pixel. Taking the product of these two values emphasizes the need for a peak to have both a large basin of attraction as well as large magnitude. The 200 peaks with the highest salience are kept and the rest are discarded.

Using this method of Peak Detection, the computer can find single points that represent the areas with high potential of being true geological features. A semblance velocity image with computer-detected peaks superimposed on the image is given in figure 4. The circles in this image represent the relative size of the basins of attraction, while the gray scale represents the semblance velocity magnitude (the lightest areas have highest magnitude).

4. Curve Selection

Curve selection is cast as the problem of finding a subset of peaks that optimize a heuristic criterion function, in our case minimizing an error function E. In this way, the task is related to others we have considered before, such as optimal matching of 2D line segment models to cluttered and complex line data [4, 3], matching 3D line models to 2D image features assuming 3D perspective projection [1, 2, 8], optimal matching of 3D models to multi-modal

data [11], and recent advances in combinatorial line matching using local search within genetic algorithms [14].

Formally, let $\mathcal{Q}$ be the set of n peaks extracted from the Semblance Velocity image by the procedure described above. The best solution to the velocity picking problem is a subset s^* of peaks that minimizes E:

$$E(s^*) \leq E(s) \quad \forall s \in 2^{\mathcal{Q}} \tag{1}$$

In other words, the best velocity pick is the set s^* from the 2^n elements of the powerset of $\mathcal{Q}$.

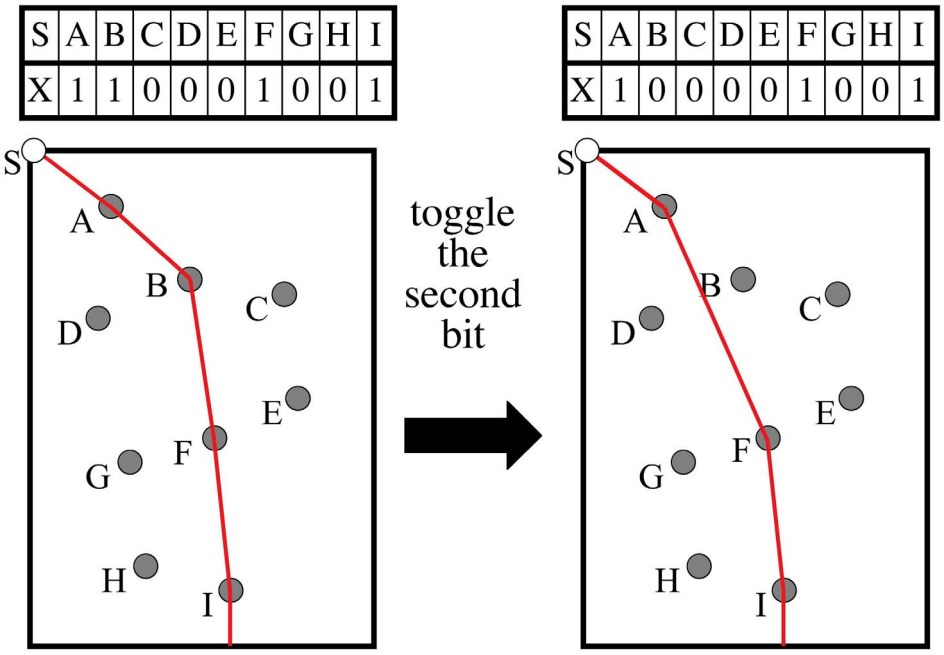

Figure 2. *Illustration of Curve Encoding and Local Search Neighborhood.*

Every subset $s \in 2^{\mathcal{Q}}$ specifies a unique curve. The mapping from peaks to curve can be explained as follows. Take all the peaks in s and sort them by vertical placement in the image. In other words, order the peaks from top to bottom. Place at the head of this sorted list of peaks a fixed starting peak at the top of the image. This peak is provided to us, and represents the semblance velocity at the earth's surface. The curve is a polyline passing from point to point in this ordered list. In order to carry the curve all the way to the bottom of the image, a final vertical segment is appended to the end of the polyline going from the lowest peak to the bottom of the image.

It will simplify our latter development of a local search algorithm to use a bit string to represent curves. Figure 2 illustrates this encoding. The left side of Figure 2 shows an encoding of a curve through four out of nine peaks: the peaks here are hand drawn to permit a simple illustration. The starting point is clearly evident at the top left

as is the final vertical segment from the last peak to the bottom of the image. The right hand side of this figure shows how toggling a single bit generates a new curve. These toggles are the basis for the local search algorithms described below.

Soft constraints, such as the desire to have the curve pass over regions of high semblance velocity, are encoded in the error function E. In addition, hard constraints are introduced that preclude some subsets of peaks. The specific hard and soft constraints used are described in Section 4.1.

Several different heuristic search algorithms have been developed to search the combinatorial space of possible curves. These are all local search variants with a neighborhood defined by the operation of adding or removing peaks from the current curve. They are described in section 4.2.

4.1. Error Terms

A variety of different possible soft constraints were encoded as errors and tested as part of a composite error term E. Three terms were found to be most important. The function is designed to be minimized.

$$E(s) \;=\; w_1 E_{en}(s) + w_2 E_{an}(s) + w_3 E_m(s) \tag{2}$$

The specific curve being evaluated is represented by the bit string s. Assuming there are m peaks, i.e. m 1's, in s, and k edges going from the top to the bottom of the velocity image, then the three terms are defined as follows:

Inverse Energy: (E_{en}) The total energy under the curve is one measure that has geologic significance. This inverse energy error is defined as:

$$E_{en}(s) = 1 - \left(\frac{1}{\sum_{k=1}^{m} \ell_k}\right) \sum_{k=1}^{m} E_{en}(s,k) \tag{3}$$

$$E_{en}(s,k) = \ell_k \frac{\sum_{x=x_{min}}^{x_{max}} \sum_{y=y_{min}}^{y_{max}} w_{k,x,y} v_{x,y}}{\sum_{x=x_{min}}^{x_{max}} \sum_{y=y_{min}}^{y_{max}} w_{k,x,y}} \tag{4}$$

and

$$\begin{aligned} x_{min}, y_{min}, x_{max}, y_{max} &\equiv \text{bounding box for edge } k \text{ padded by } 2 \\ \ell_k &\equiv \text{length of edge } k \end{aligned}$$

$$
\begin{aligned}
w_{k,x,y} &\equiv \max\left(0, 1 - \frac{d_k(x,y)}{2}\right) \\
v_{x,y} &\equiv \text{velocity at } x,\ y \text{ in range } [0,1] \\
d_k(x,y) &\equiv \text{distance from } x,\ y \text{ to edge } k
\end{aligned}
$$

To summarize, for each edge k, compute a weighted sum of the inverted normalized semblance velocities within 2 pixel units of the edge. The weighting is linear: decreasing with distance to the edge. The total for all edges is a sum weighted by the length of each edge and then normalized by the sum of the weights.

Average Turning Angle: (E_{an}) This is the average of the squares of the turning angles in the polyline. Smooth polylines have low E_{an}.

$$E_{an}(s) = \frac{\tau}{m} \sum_{k=1}^{m} (\theta_{k-1} - \theta_k)^2 \tag{5}$$

where

$$
\begin{aligned}
\theta_k &\equiv \text{angle of edge } k \text{ in degrees} \\
\theta_0 &\equiv \text{angle of phantom edge entering image} \\
\tau &\equiv \text{normalization constant}
\end{aligned}
$$

There is a phantom edge 0 to which the first edge is connected. This edge represents the velocity at the surface. Typically a value of $\theta_0 = 22.5$ degrees is used. The normalization constant brings the values of the error within the range zero to one. Based upon experience using the velocity picking system in an iterative mode, $\tau = \frac{1}{3000}$ has been selected.

Proximity to Median: (E_{me}) The area between the curve and a median curve that acts as an exemplar for this region of the seismic survey. More precisely,

$$E_{me}(s) = \gamma \sum_{r=1}^{t} \|x_{s,r} - x_{m,r}\| \tag{6}$$

where

$$
\begin{aligned}
t &\equiv \text{the number of rows in the image} \\
x_{s,r} &\equiv x \text{ coordinate of current path at row } r \\
x_{m,r} &\equiv x \text{ coordinate of median path at row } r \\
\gamma &\equiv \text{normalization constant}
\end{aligned}
$$

The median is generated automatically, as described below. Based upon experience using the system in an interactive mode, $\gamma = \frac{1}{3000}$ has been selected.

To make computing $E_{en}(s)$ during search efficient, we pre-compute $E_{en}(s,k)$ from equation 4 for all possible edges. During search, we sum up the pre-cached Energy-Length products and divide by the sum of the lengths for the edges in the curve. The resulting value represents the inverse of the average intensity under the curve.

Were we to minimize $E_{en}(s)$ only, highly implausible solutions from a geological perspective might be favored. The angle error, $E_{an}(s)$, expresses the additional constraint that semblance velocity varies smoothly. Another constraint is that velocity must monotonically increase as the curve progresses from the top to the bottom of the image. This constraint always holds for real geology–except in very unusual circumstances. In terms of the edges out of which the polyline is assembled, no valid segment can slope to the left: the lower end point is either directly below or to the right of the upper end point. This hard constraint dramatically reduces the number of possible curves. However, it does not fundamentally alter the fact that the number of possible curves can grow exponentially as a function of the number of peaks n.

4.2. Local Search Algorithms

As already suggested by Figure 2, during the search for a best subset of peaks, a solution is encoded as a string whose length is equal to n. However, Figure 2 was a slight over simplification. Each cell in the string can take on one of four values: ON, OFF, $ALWAYS$, or $NEVER$. The ON and OFF correspond to 1 and 0 in the bit string encoding shown in Figure 2: ON means the peak is in the curve and OFF that it is not.

The other two values allow a user to further interact with the system in directing search for the best curve. Specifically, the user can interactively indicate that a peak must $ALWAYS$ or $NEVER$ be included in the solution; the peaks marked as $ALWAYS$ or $NEVER$ are not manipulated during search. The results reported

later in the paper were generated in a fully automated search. In other words, $ALWAYS$ and $NEVER$ were not used. However, in demonstrations of the system, prospective users reacted very favorably to the extra degree of control.

After a great deal of experimentation, we ended up using rather simple local search methods for search. The essential operation is that of toggling all of the n peaks to generate a local neighborhood of alternative solutions. Toggling simply means that if a peak is part of the curve, then it is removed, and if a peak is not part of the curve, then it is added. After testing the n neighbors of the current solution, the one with the greatest drop in $E(s)$ is selected as the new solution. Search stops when none of the n neighbors is better. This is essentially a steepest ascent hill climbing search.

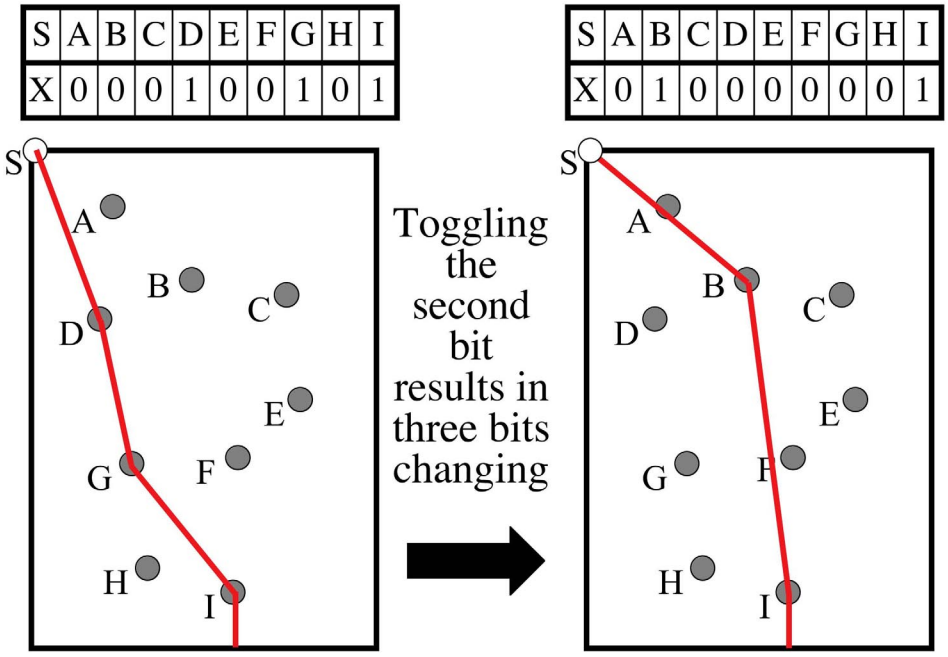

Figure 3. *Toggling one bit can result in others being changed.*

At first appearance, this would seem to be a search of all bit strings differing by only one bit, i.e. a Hamming distance one neighborhood. However, the constraint that velocity must increase monotonically alters the neighborhood structure. In particular, flipping a bit can sometimes have side effects. For example, when a peak is turned on, all peaks above and to the right of it must be turned off, as must all peaks below and to the left of it. This constraint keeps the string within the set of valid solutions. Since the addition of a peak can cause other peaks to be removed, the space of curves is connected in a way that allows neighboring strings to differ by more than one

bit.

This connectivity is illustrated by Figure 3. In this example, adding the second peak, B, causes peaks D and G to be removed in order to prevent the curve from doubling back to the left. Thus, in this example toggling one bit results in a move of three bits in Hamming distance. What we see here is that the monotonicity constraint not only helps us by reducing the total number of solutions, but also by allowing the neighborhood to includes jumps of many bits.

In practice we use three variants of the basic steepest ascent algorithm:

Zero Single Search All bits are set to OFF, and a single pass of hill climbing is applied. This search works in a similar way to what is known as a "greedy tree search." The polyline can be thought of as a tree, where the root of the tree is the first peak that is selected. Note that when this first peak is selected, the problem is then decomposed into two parts: 1) the set of peaks in the top half of the image and 2) the set of peaks in the bottom half. The algorithm adds or removes the peak yielding the greatest improvement at each decision. This process continues until no further improvement can be made. In effect, this is a single pass bit climber that starts from the string of all zeros and climbs to a local optimum.

Current Single Search This search is identical to the Zero Single, except that the string is not first set to OFF as it is in the Zero Single Search. "Current Single Search" is used by the user to search the neighborhood surrounding the current candidate solution for improvements. The start state of the search can be defined by the user or randomly selected.

Multiple Search Search begins with a Current Single and a Zero Single search, followed by j trials of random restart local search. For these j trials, solutions are generated at random and then hill climbing is applied to each. The best solution is then chosen from the resulting set of $j + 2$ locally optimal solutions.

In practice, we have found Zero Single Search generates solutions closer to those generated by humans. Even though Multiple Search can sometimes find curves with lower $E(s)$, in the worst case these are very far from the human pick. This discovery that finding the most optimal curve as defined by $E(s)$ can actually lead away from solutions favored by human experts leaves us in the short term enjoying the happy accident that the cheap algorithm does what we want better than an expensive one. However, it also suggests further study of the criterion function may be in order.

In an ideal world, our criterion function would always be an accurate reflection of human preference. Were

this the case, it would be worthwhile to consider alternative optimization procedures for finding globally optimal curves. Our initial decision to emphasize local search was in part associated with our taking a very generic approach to the search space. For example, initially we did not anticipate the monotonicity constraint. Given this constraint, there are other approaches to consider. These include constraint based tree search and linear programming.

4.3. Energy and Angle Alone are Inadequate

Using either the Zero Single or the Multiple Search algorithms to optimize the first two terms in equation 2, $E_{en}(s)$ and $E_{an}(s)$, is problematic. More precisely, the majority of the time the algorithm finds excellent solutions relative to the human picks. However, a minority of the time, the solutions deviate widely from the human picks. This caused us months of consternation, and we tried many variants upon the error function and the search algorithm.

We went so far as to develop a genetic algorithm to search a space of parameterized error functions. This parameterized error function included many other soft constraints that we thought might enable the algorithm to reliably identify good curves. Using the genetic algorithm, each member in the population represented a different error function parameterization. The fitness function for the genetic algorithm was related to how closely the resulting curves matched human picks. We let the genetic algorithm search for days at a time, and in the end we found no combination where better than perhaps 75 to 80 percent of the problems were solved in a manner consistent with the human picks.

As we were pondering the problem of finding a combination of search algorithm and error function that would be highly reliable, the salient aspect of the situation began to sink in: most of the time the Zero Single algorithm was getting very close to the human picks. This put us in the mind to think about robust statistics, and in particular, the median. This thought, combined with a desire to constrain search on a particular image to search near an exemplar derived from the local geology, led us to the algorithm presented in the next section.

4.4. Searching Near the Median

The Median Search algorithm is our most successful algorithm. It relies upon two assumptions. First, the velocity between images should not differ greatly, since all of the images in a given set are taken from approximately the same geographical site. Second, using the two earlier error terms (Energy Error and Angle Error) alone, a zero

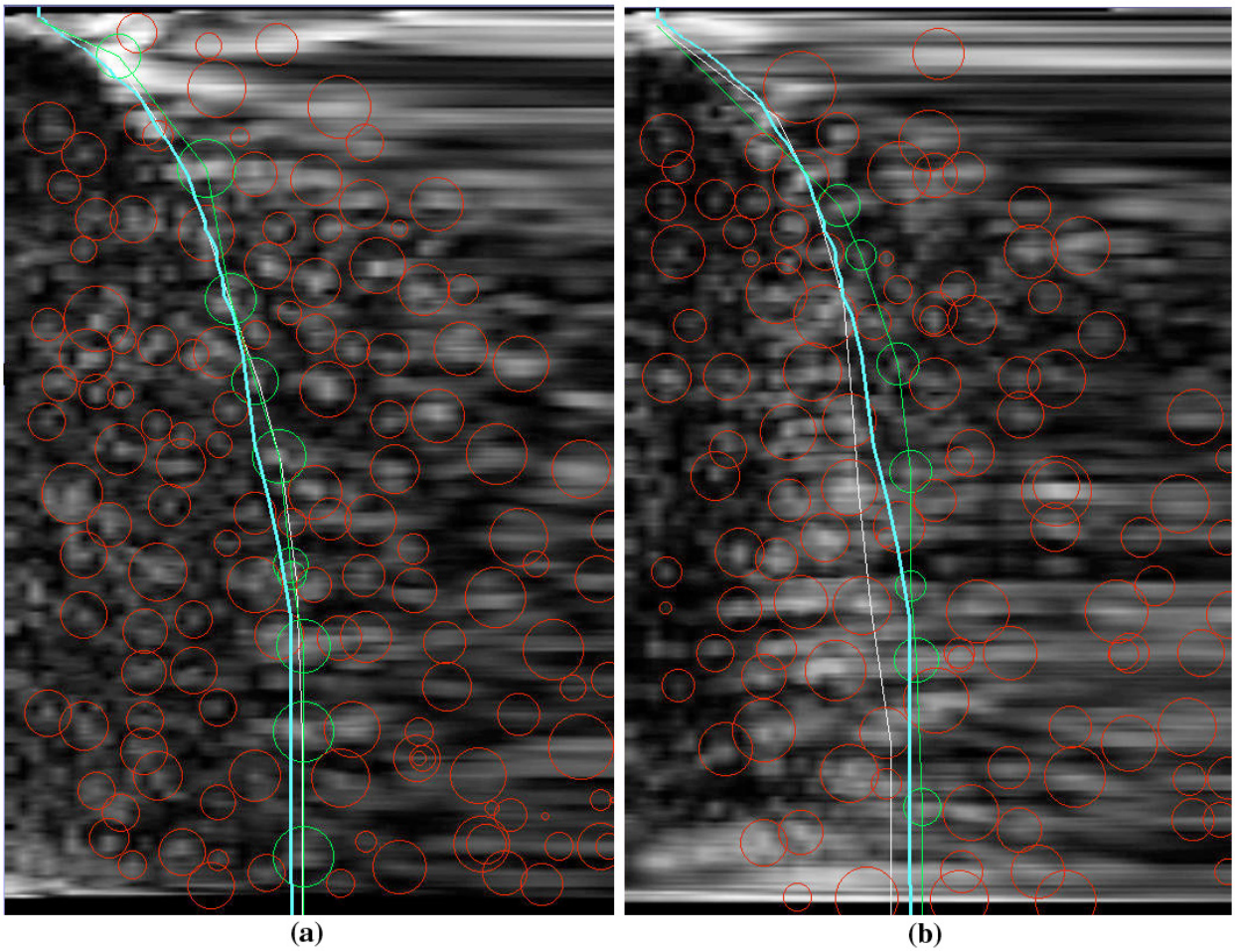

Figure 4. Sampling Solutions on a very hard dataset. a) Automated pick is essentially equivalent to human pick. The fat median line lies to the right of the hand picks and automated picks, which are nearly identical. b) Automated pick deviates from human pick. The hand picks are leftmost, the automated picks are rightmost and the median falls in the middle.

single search will result in a reasonable solution the majority of the time. When it does fail, the resulting solution is clearly an outlier.

To begin, a Zero Single search is performed on a each of a set of i images associated with approximately the same geographical site. The result is i separate semblance velocity picks and the associated polyline connecting the peaks. Each of these solutions specifies a particular velocity at a particular time. Therefore, at any given time we have i velocities. For each time value, the median of these i values is computed and a new artificial median semblance velocity polyline is plotted through the data. This median is shown in blue Figure 4.

After the median curve is computed, $E(s)_{me}$ in equation 2 can now be computed. As already noted, its value

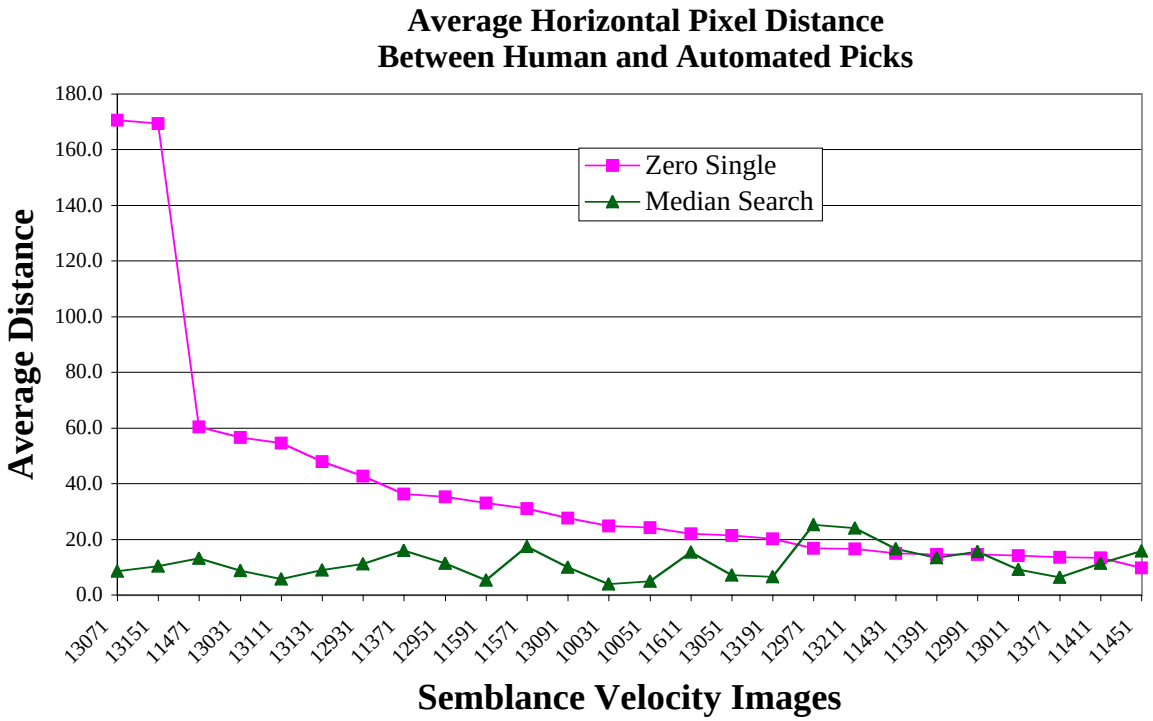

Figure 5. Squared area between human and curve and curves found using our algorithms. Data for 26 images is shown, sorted by decreasing area for the Zero Single algorithm.

is proportional to the area between the polyline being evaluated and the median polyline. The use of this error term adds continuity to the solutions in the set. By using this median polyline as an exemplar, but still allowing the program to search each image individually, a trend can be preserved while still exploiting the variations in each image. The Median method can be repeated in an iterative fashion in order to refine the median if necessary. In practice, a single refinement after the initial median calculation dramatically improves the result, and further refinements are unnecessary.

5. Results

Landmark Graphics provided us with 26 semblance velocity images [1]. For each, a human interpreter generated a semblance velocity curve. This allowed us to compare our automated picks with the human picks, both visually and numerically. A good measure of how closely our curve matches the human pick is average horizontal distance between the two curves.

[1]We regret we can not present results on more data. Our initial research plan called for a much more extensive evaluation, but for reasons having nothing to do with the technical merits of the project, funding for the work was cancelled prior to the final evaluation.

Figure 5 shows this measure for the Zero Single Search and the Median Search algorithms. It shows the Median algorithm is generating curves much closer to the human pick. For both algorithms, the weights in equation 2 were set to 1.0, 1.0 and 0.5 respectively. Two things are evident from this plot. First, the Median algorithm is finding solutions much closer to the human picks. Second, about half the Zero Single solutions are quite good, hence the justification for the median serving as an exemplar.

Two example velocity picks, curves, for the Median algorithm are shown in Figure 4. This Figure also shows the peaks, the median pick and the hand generated pick. The peaks in red are not part of the Median algorithm's pick: those in green are part of the curve. The hand picked solution is shown as a white line. The automated pick is shown as a green line. The median is a fat blue line.

The image in Figure 4a, image 13111, exemplifies the Median algorithm doing very well. The automated pick is, for all intents and purposes, as good as the hand pick. The area between the automated and hand pick for this image is $4,399$: this is one of the best in the set. The image in Figure 4b, image 12971, is the worst case for the Median algorithm. The area between the automated and the hand pick for this image is $19,755$. Even this automated pick is plausible.

It is striking that the median is so frequently close to the hand picks. The distance between the polylines representing the median and the line associated with the hand pick velocities is usually *less than* the distance between the individual automated solutions and the line associated with the hand picked velocities. This could be interpreted as evidence that using the median is better than using the individual solutions since it is closer to the hand-picked solutions. But median does not maximize the energy in each individual semblance image.

In fact, the energy associated with the automated solutions is greater than the energy associated with the hand picked solutions. Thus, it is possible the automated solutions are better than the set of hand picked solutions. A geophysicist–perhaps even one familiar with the local geology represented in these images–would have to make that determination.

6 Conclusion

We have presented a novel and domain specific solution to the task of velocity picking. The solution is tested relative to velocity picks made by a human expert on a test data set of 26 images. The algorithm we developed first identifies peaks of high intensity in the Semblance Velocity image, and then uses a local search procedure to find a locally optimal subset of peaks that represent the most coherent curve through the image. The two step process,

identification of peaks followed by connecting of peaks to form a best curve (polyline), mimics the manner in which geophysicists perform the task.

The most significant aspect of the paper is that our algorithm has performance comparable to that of the human expert. We know of no other automated solution to this task. The apparent simplicity of the solution presented should not confuse the reader into thinking this was a simple problem. On our way to the solution presented we tried many more complicated and involved algorithms. All of which performed poorly.

The most original idea in the paper is the use of a median solution over a set of related images to provide an additional constraint for the solution in each image. This two phase search approach, where the median of the result of the first phase guides the solutions in the second phase, was essential to our solving this problem. Moreover, while much we have presented is highly domain specific, the idea of using a median shape as an exemplar will generalize to other tasks.

References

[1] J. R. Beveridge. *Local Search Algorithms for Geometric Object Recognition: Optimal Correspondence and Pose*. PhD thesis, University of Massachusetts at Amherst, May 1993.

[2] J. R. Beveridge and E. M. Riseman. Optimal Geometric Model Matching Under Full 3D Perspective. *Computer Vision and Image Understanding*, 61(3):351 – 364, 1995.

[3] J. R. Beveridge, E. M. Riseman, and C. Graves. Demonstrating Polynomial Run-Time Growth for Local Search Matching. In *Proceedings: International Symposium on Computer Vision*, pages 533 – 538, 1995. IEEE PAMI TC, IEEE Computer Society Press.

[4] J. R. Beveridge, R. Weiss, and E. M. Riseman. Combinatorial Optimization Applied to Variable Scale 2D Model Matching. In *The IEEE International Conference on Pattern Recognition 1990*, pages 18 – 23. IEEE, 1990.

[5] D. M. Nguyen. An iterative Technique for Target Detection and Segmentation in IR Imaging Systems. Tech Report, (CECOM) Center for Night Vision and Electro-Optics, 1990.

[6] B. Fish. A Neural Network Approach to Automate Velocity Picking. In *Proceedings of SEG* 64^{nd} *Annual International Meeting*. Society of Exploration Geophysicists, 1994.

[7] P. Fua and Y. G. Leclerc. Model driven edge detection. In *Proceedings: Image Understanding Workshop – 1988*, pages 1016 – 1021. DARPA, Morgan Kaufmann, April 1988.

[8] J. R. Beveridge, E. M. Riseman, and C. Graves. How Easy is Matching 2D Line Models Using Local Search? *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(6):564 – 579, 1997.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *First International Conference on Computer Vision Proceedings*, pages 259 – 268, 1987.

[10] M.R. Stevens and J.R. Beveridge. Optical Linear Feature Detection Based on Model Pose. In *Proceedings: Image Understanding Workshop*, pages 695–697, 1996. ARPA, Morgan Kaufmann.

[11] Mark R. Stevens and J. Ross Beveridge. Precise Matching of 3-D Target Models to Multisensor Data. *IEEE Transactions on Image Processing*, 6(1):126–142, January 1997.

[12] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*, chapter Local Search, pages 454 – 480. Prentice–Hall, Englewood Cliffs, NJ, 1982.

[13] D. Whitley, J. R. Beveridge, and C. Ross. Automated velocity picking: A computer vision and optimization approach. Technical Report CS-98-114, Computer Science Department, Colorado State University, 1998.

[14] D. Whitley, R. Beveridge, K. Mathias, and C. Graves. Testing Driving Three 1995 Genetic Algorithms. *Journal of Heuristics*, 1:77–104, 1995.