

Contents

- Presentation Aspects of VIZ 2
 - Viz Graphical User Interface 2
 - Main Features 2
 - Multi-Source Visualizations 6
 - Learning via Visual Comparison 6
 - Animated Search Playback 8
 - Color and Highlighting 9
 - Multiple Level of Details 11
 - Text-Based Information Center (TBIC) 12

Presentation Aspects of VIZ

We now discuss various presentation aspects of VIZ which are essential for effective explanation of search behavior.

Viz Graphical User Interface

Main Features

VIZ is a local search visual analysis suite embedded with many features. The user just need to log certain information (straightforward) from their local search algorithm. The computation/analysis is done by VIZ. We hope that with VIZ alone, the algorithm designer can get almost all that he/she needs to know about his/her algorithm...

VIZ features are:

1. The key visualization is animation of search trajectories (applicable to any trajectory-based metaheuristic search algorithms such as TS, ILS, SA, etc) on abstract 2-D space which is applicable to arbitrary local search algorithms — [this is our new concept]. This is combined with other supporting visualizations ranging from generic ones to problem and algorithm specific visualizations. See section ?? + ??.
2. Visualizations in VIZ are designed to follow certain visualization guidelines, like the one shown in [4, 5, 6]. VIZ GUI also follow the strict standard of Windows UI design guidelines set by Microsoft (cite MS UI standard, etc) and UI book (cite DTUI, Schneiderman).
3. Multi platform as the interface to local search is via a very simple log file format. The hard computations will be done by VIZ.
4. Customizable

Overall

To make a visualization tool that is easy to use and quick to learn, the design of the GUI is of fundamental importance. In this section, we justify several design choices of VIZ GUI (see Figure 1)¹.

VIZ is developed using Microsoft Visual C# 2005 which utilizes .NET Framework 2.0² for the implementation of the common controls used in Windows XP. The core visualizations screens are drawn using CsGL: a C# Graphics Library wrapper for OpenGL, the standard Graphics Library originally written for C/C++.

The GUI elements should be familiar as it utilizes the Windows XP look and feel and borrows features from other well-known applications.

VIZ used various computer graphics technologies, e.g. alpha blending to reduce visual clutter and smooth animation, highlights and shading.

VIZ also automatically generate index points (points that are deemed to be interesting in the recorded search, e.g. local optima) for playback. (see subsection).

Ability to compare the behavior of two local search algorithms (see subsection).

Interaction: Menus, Toolbars, and Status Bar

Users can interact with these GUI elements via a pointing device/mouse (primary interaction mode) or via keyboard (there are plenty of keyboard shortcuts for common functions).

We design the search playback controller (play/pause/stop with speed controller) to emulates the look and feel and behavior the controllers of commonly used media player programs, e.g. Windows Media Player (see Figure 2). The major difference is that in VIZ, we have *two* playback controllers which can be operated independently or simultaneously/linked (see subsection ?? for details).

¹According to the definition of [2], VIZ can also be classified as a ‘dashboard’ for information visualization.

²We are aware that developing VIZ using .NET Framework will force VIZ to solely run in Windows-based PCs. However, VIZ is an independent tool, as it can still analyze the local search algorithms which are implemented in other platform by communicating via text-based log files.

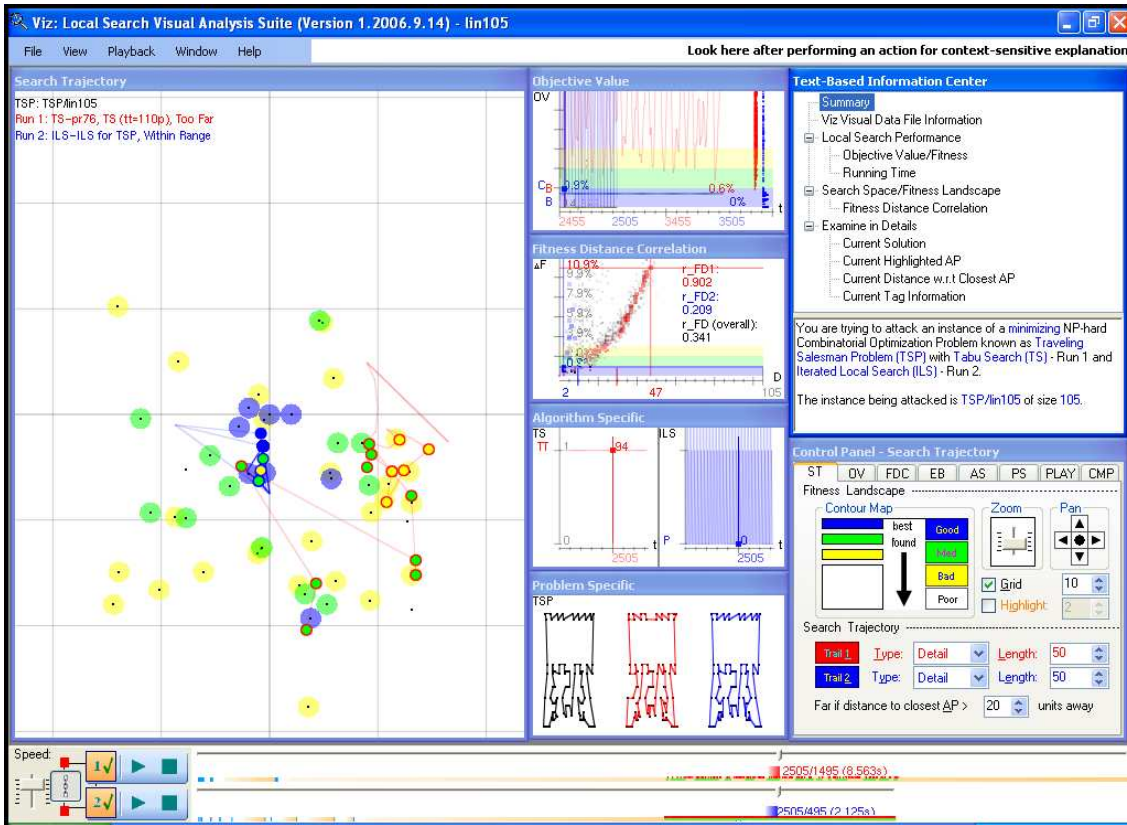


Figure 1: Viz GUI

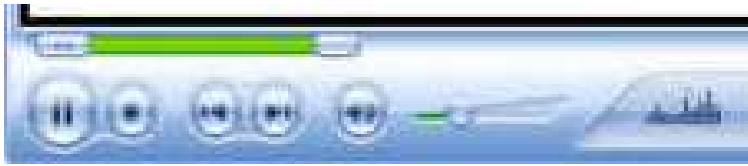


Figure 2: Windows Media Player Playback Toolbar

Status bar are placed on top rather than the usual position in the lower part of the screen. We do this to maximize screen real estate. VIZ GUI is already complex and we need to save screen space. Therefore, the space on the right side of menu is used instead.

We design the menu structure (see Table 1) including its shortcuts are designed to be as similar as possible with common windows-based applications.

Note: 1-7 is also accessible via search playback toolbar.

Menu	Shortcut	Description
[F]ile -[N]ew Viz Visual Data File Wizard... -[O]pen Viz Visual Data File... -[L]oad Default Settings -[S]ave Settings into Viz Visual Data File -[S]ave Screen-Shots -[E]xit	Alt+F Ctrl+N Ctrl+O Alt+F,L Alt+F,S Ctrl+Shift+S Alt+F,X	Starts the Viz Raw-to-Visual Data Conversion Wizard! A confirmation screen will pop up. Click ‘yes’/‘no’ accordingly. No direct shortcut as Ctrl+S is taken by ‘Stop’. Screenshots are saved in the ‘/Screen Shots’ folder. Exit Viz
[V]iew -[S]how/Hide: -[S]earch Trajectory -[O]bjective Value -[F]itness Distance Correlation -[E]vent Bar (always on) -[A]lgorithm Specific -[P]roblem Specific -[C]ontrol Panel -[T]ext Based Information Center -ST [F]itness Landscape Overview -ST [S]earch Coverage Overview -ST Search Trajectory [D]etailed View -OV [O]verview Analysis -OV [D]etailed Analysis -Contour Map: [U]niform -Contour Map: [N]ormal -Contour Map: [R]ugged -[C]olor Scheme -[P]rinted Paper -[C]omputer Screen -Visual Comparison [M]ode -[R]efresh -[F]ull Screen	Alt+V Alt+V,S Ctrl+1 Ctrl+2 Ctrl+3 Ctrl+4 Ctrl+5 Ctrl+6 Ctrl+7 Ctrl+8 Ctrl+Shift+Q Ctrl+Shift+W Ctrl+Shift+E Ctrl+Shift+R Ctrl+Shift+T Ctrl+Shift+Z Ctrl+Shift+X Ctrl+Shift+C Alt+V,C Alt+V,C,P Alt+V,C,C Alt+V,M F5 F11	User can decide which visualization to show/hide As this is not a window, we can’t hide it. Good for fitness landscape analysis. Good to show overall search coverage. This is the commonly used trajectory analysis mode. Condense the OV chart to fit into one window. OV normal view, view the fluctuation iteration by iteration. Good quality, small spread. Normal quality, normal spread. Many AP are not good, big spread. White background. Blue/soothing background. Shows an overview of the comparison modes of all visualizations. Refresh/redraw all the visualizations. Take the whole screen space and maximize all child windows.
[P]layback -[1]st Search Trajectory ¹ : Active -[2]nd Search Trajectory ¹ : Active -[L]inked ² : Yes -[Fo]cus ³ : Both -[P]lay/Pause ⁴ -[S]top ⁵ -[F]orward ⁶ -[B]ackward ⁶ -[F]aster ⁷ -[Sl]ower ⁷ -Edit [T]ag Information	Alt+P Ctrl+Alt+1 Ctrl+Alt+2 Ctrl+F Ctrl+P Ctrl+S Ctrl+Right Ctrl+Left Ctrl+Up Ctrl+Down Alt+P,T	Options: Active/Not Active. Options: Active/Not Active. If linked, both trajectories’ play/pause/stop/etc are linked. Set focus of the playback command: 1st, 2nd or both. Playback the active Trajectory(ies). Stop playback of active Trajectory(ies). Advance active Trajectory by X iterations/Y milliseconds. Rewind active Trajectory by X iterations/Y milliseconds. Set playback speed 50ms faster. Set playback speed 50ms slower. Open the tag editor window.
[W]indow -[S]earch Trajectory only -[O]bjective Value only -[F]itness Distance Correlation only -[A]lgorithm Specific only -[P]roblem Specific only -[B]asic Window Layout (All Visualizations) -[W]ithout Algorithm & Problem Specific	Alt+W Ctrl+Shift+1 Ctrl+Shift+2 Ctrl+Shift+3 Ctrl+Shift+5 Ctrl+Shift+6 Ctrl+Shift+7 Ctrl+Shift+8	This menu contains several built-in window layout modes.
[H]elp -[E]xplain: -[Q]uestions about Local Search Behavior -[V]iz Raw-to-Visual Data Conversion [W]izard -[V]isualizations in Overview -[S]earch Trajectory -[O]bjective Value -[F]itness Distance Correlation -[E]vent Bar -[A]lgorithm Specific -[P]roblem Specific -[C]ontrol Panel -[T]ext-Based Information Center -[V]isit Viz’s Website -[R]ead Me First -[V]iz Version [H]istory -[A]bout Viz	Alt+H Alt+H,E Alt+H,E,Q Alt+H,E,W Alt+H,E,V Alt+H,E,S Alt+H,E,O Alt+H,E,F Alt+H,E,E Alt+H,E,A Alt+H,E,P Alt+H,E,C Alt+H,E,I Alt+H,V Alt+H,R Alt+H,H Alt+H,A	A good tool must come with a detailed information.

Table 1: Explanation of Viz Menus — Not up-to-date

In VIZ, right clicking on various parts of the screen reveals context-sensitive menu. For example see Table 2.

Menu	Shortcut	Description
-[E]xplain this Visualization	RightClick	Similar as Help-Explain-[the window].
-[H]ide this Window	RightClick	Similar as View-Show/Hide-[the window].
-Switch to [J]uxtapose/[S]uperimpose Mode	RightClick	For quick exchange between the two comparison modes.

Table 2: Explanation of VIZ Context-Sensitive Menus — Not up-to-date

Control Panel

One notable GUI feature of VIZ is a trivial but yet very important feature: customizability/adaptability, as not every user are comfortable with the default visualization settings. The control panel in VIZ allows user to adjust various array of options, such as to adjust various level of details of VIZ visualization screen, to adjust color and to toggle various highlights, to choose between juxtapose or superimpose visual comparison mode, etc.

This control panel, other than as customization tool, is also acts as a filtering/highlighting tool. This interaction options provided in the control panel is our way to implement the Information Visualization mantra: “Overview first, Zoom and Filter, then Details on Demand”.

We build a separate Control Panel window rather than embedding each visualization options directly in the visualization screen for one sole reason: to save screen space.

The tab page is selected in context depending on the current active visualization window.

The complete list of customization options are listed in the respective sections throughout this chapter. Customizing the way a visualization looks may render important insights, e.g. compacting the OV chart allows us to do RTD analysis

Due to the limited screen real estate, we design the control panel to be as compact as possible — but with proper groupings to avoid confusion. Moreover, apart from default values, we save user customization within the individual VIZ visual data file so that whenever the user re-load the VIZ visual data file in the future, he/she will see exactly the same visualization as before — the settings, the windows layout, etc, persist beyond the execution of VIZ.

Coordinated Visualization Screens

VIZ makes use of multiple visualizations which are animated *in concert* via a VCR-like time controller to graphically playback the previously recorded search. See subsection .

The description of these visualizations have been explained in the previous subsection.

Using the Power of Data Analysis

In addition to multiple visualizations, we also have text-based information center to display information that are better displayed textually. This text-based information center is designed similar to property boxes commonly found in applications (tree-view to show categorical information and the detailed view in another display) (e.g. Windows Explorer, Microsoft Outlook).

Help System

Acceptance of software system often decided by its help system. We want VIZ to be widely applicable by many users. So, this part is not left out...

We also believe that in order for quick understanding of a GUI, enough assistance should be given whenever user needs help. In VIZ, such help/assistance is everywhere. For all adjustable range values, we give a suggestion or range of possible values whenever the user hover his/her mouse over the input controller, for example: AP Size: [10..1000], default values: 50, etc. Virtually, for all GUI elements, pressing F1 while focusing on that GUI element will display contextual popup help about the feature/functionality of that GUI element. Hovering your mouse will show tool tips.

Error messages are minimized, we adopt error prevention GUI. If necessary, error messages are displayed in non-intrusive manner.

Various other design choices will be discussed in the next subsections, e.g. choice of MDI layout.

Multi-Source Visualizations

Explanations

Each visualization that has just been described in the previous section is capable of explicating some aspect of local search behavior which may have been hidden in the search trajectory. However, relying on one type of visualization *alone* can be myopic and does not convey the full picture of what is happening during search. We believe that multiple points of view³, which are designed to move simultaneously *in concert*, are required given the difficulty of analyzing local search behavior.

We have several examples of the importance of having multi-source visualizations:

1. We could observe solution cycling in ST visualization but if we do not also observe the AS visualization, we may not realize that the cause of such cycling is because of the low tabu tenure at that moment.
2. We can use event bar's highlights to move into the iterations where certain generic events are occurring, and then look at the ST visualization to see what is actually happening, or to verify that improving step in OT visualization.
3. We see a pattern of 'big valley' in ST visualization, and we verify its existence in FDC visualization.
4. EB and OV visualization are scaled using the same scale and drawn side by side (top bottom).
5. We can verify perturbation behavior (e.g in ILS, in strong diversification phase of TS) in the PS visualization.

In psychology, Situation Awareness theory [1] points out that the human user is unable to see multiple visualizations if all require high attention at the same time. When human is bombarded with a number of displays, the overall scanning ability drops resulting in concentration on only a small fraction of the displays, thus losing Situation Awareness.

To address this problem in VIZ, we designate the search trajectory visualization as the main screen where most information about local search behavior will be displayed, this visualization will be the focus of the user's attention. The other visualizations serve as peripheral visualizations to backup the main one.

Nevertheless, as search playback in VIZ is *not on-line*, this issue is not so severe as the user can always pause, rewind, and replay the search at any time in case some information was missed. The underlying visualization information doesn't change with time.

To better support such multi-source visualizations, VIZ presents the coordinated visualization windows in a Multiple Document Interface (MDI) style.

The choice of MDI layout conform with Situation Awareness theory in reducing the information overload as the user has the freedom to show, hide, scale, or layout the visualization child window(s). User can concentrate on the central visualization and recall the peripheral window(s) only when needed. This is hard to be emulated using one single window which displays all visualizations at one time (Single Document Interface — SDI).

Learning via Visual Comparison

Explanations

Human is not a 'good judge' as human is not good in judging an absolute quality of an object. However, human is a creature comparison as human is good in visually comparing objects. When given a baseline with known value, we can compare this baseline with other object X . Using human strength in detecting visual similarities and differences, human can distinguish certain parts of these objects and tell the relative quality of X w.r.t the baseline.

Knowing this fact, we included an interesting feature into VIZ: the ability to learn local search behavior via visual comparison. This is done by allowing one to playback two local search runs concurrently and draw both visualizations in either juxtapose (side-by-side) or superimpose (overlap) mode. This feature is available in almost visualizations screens.

There are various modes that we can utilize.

³We may be able to cramp all individual visualization together into one 'big' visualization. However, it will be very convoluted and thus not effective. Multiple views of the same thing from different angle is actually helpful to reduce such complexity.

1. We have two search runs of local search LS_{run1}^{ϕ} and $LS_{run2}^{\phi'}$, where LS_{run1} is LS with parameters ϕ and LS_{run2} is LS with some different parameters ϕ' . Then, we can investigate whether the changes in the parameters from ϕ to ϕ' is the cause for any significant differences in the search (small differences may be detected too).
2. To check the robustness of a stochastic local search by checking whether the performance of two runs of the same local search with the same settings LS_{run1}^{ϕ} and LS_{run2}^{ϕ} produce similar performance. Usually we don't want to see major variations in the local search performance and behavior.
3. To compare the general behavior of two distinct local search algorithms $LS - A$ and $LS - B$. Is one LS is superior than the other because of the way it explores the fitness landscape or just by luck?
4. To compare two parts of the same search trajectories $LS_{run(iter=1..100)}$ and $LS_{run(iter=200..300)}$ ⁴. What is being done by the local search in one phase compared to with another phase?

Visual Comparison Options

See Figure 3 and Table 3.

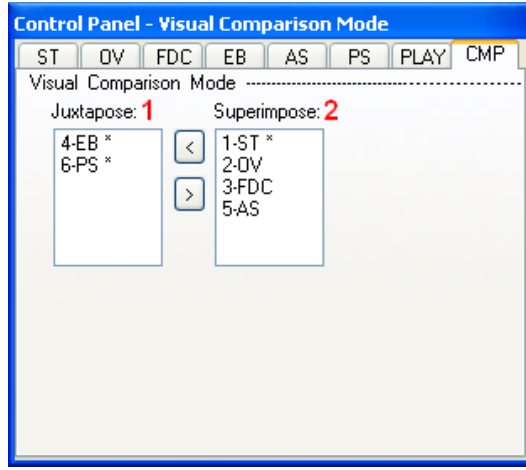


Figure 3: Visual Comparison Options.

Items	Description
1	Juxtapose: Side by side
2	Superimpose: Overlap

Table 3: Visual Comparison Options

Multi Source Visualization Options

See Figure 4 and Table 4.

Items	Description
A	You can choose which visualization window to activate/hide.
C	You can rearrange, resize, layout each of these visualization windows according to your preference.

Table 4: Multi-Source Visualizations Options

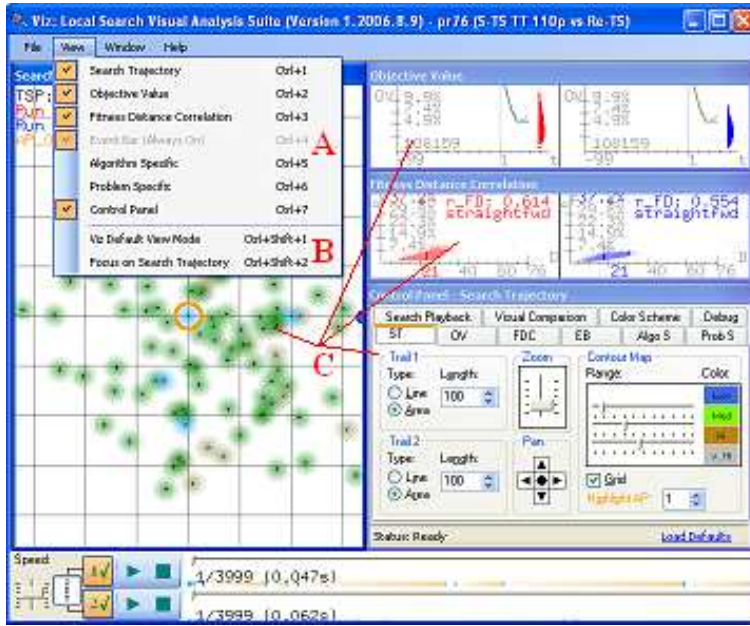


Figure 4: Multi-Source Visualizations Options.

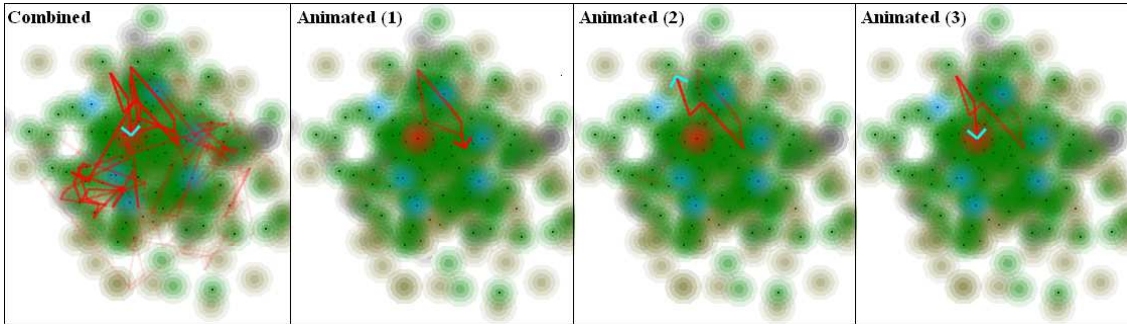


Figure 5: Combined: information is presented all at one time (useful for overview) — versus — Animated: information is presented over time (useful for details). *Alpha blending* is used to show the decaying animation (aging/showing history).

Animated Search Playback

Explanations

As local search runs for a large number of iterations, the most natural visualization is to playback the search information dynamically over time using animation⁵. Such information over time will be hard to be seen statically as trying to draw everything in one display will tend to clutter the visualization. Presenting all information at one time may be good for overview, but presenting the information over time via animation is good to display fine grained operations, which may not be visible at all in overview mode, that is, we need the temporal sequence.

A comparison of static versus animated visualization of the same data of Search Trajectory visualization is given in Figure 5. The amount of data animated is a user specifiable window of a portion of the search trajectory with l consecutive solutions. The animation makes it look like a trail of length l . By changing the window size, one can choose the tradeoff between a static and dynamic displays. Note that in VIZ all visualizations involve animation in various parts.

This animation effect is achieved by drawing a series of consecutive pictures with small changes drawn with smooth transitions. To avoid flicker⁶, VIZ exploits *alpha blending* to draw smooth

⁴By loading the same RunLog information to both RunLog 1 and RunLog 2 in the conversion wizard

⁵“Data that can’t be shown all at one time should be presented OVER time.” — Dave Collins, GUI handbook, page 30.

⁶It has been researched that the screen refresh rate should be at least 30fps for an animation to look smooth and a smooth but simple animation is preferable than complex but jerky animation!

transitions by fading out the color of the trail’s tail gradually.

VIZ also allows the user to adjust the search playback speed⁷ which determine how fast the animation will be drawn. This is essential as different individuals have different visual capacities in discerning the information from animation.

Moreover, since different local search runs usually require different running time, perhaps due to the complexity of operations per iteration, it may be unfair to playback two local search runs based on iteration only. We provide an option to playback the local search based on the ‘actual search time’.

Search Playback Options

See Figure 6 and Table 5.

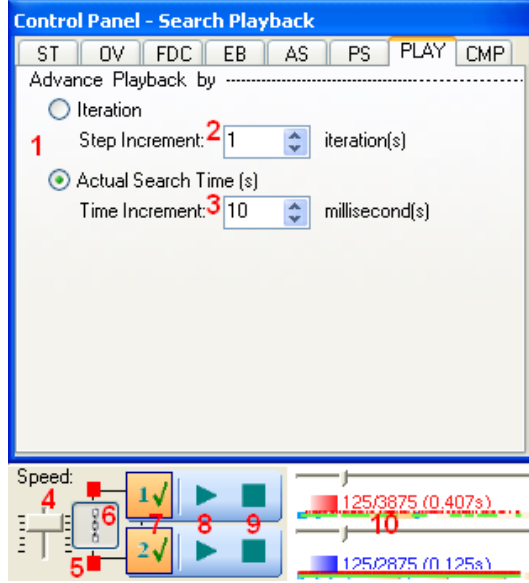


Figure 6: Search Playback Options.

Items	Description
1,3	This two options give a contrasting approach to present the same information. The standard way is comparing two playback iteration by iteration (1). The comparison using the actual search time (3) is useful when one algorithm is so slow (explore too many things) but the other are much faster.
2	This value determines the amount of iterations skipped during the playback. Since the trail is drawn backwards, there will be no gap left. This option is important to let user decide the trade off from detailed view versus overview view.
4	Similar to part B above, this value determines the amount of iterations skipped during the playback, but now by actual search time, not merely by iterations.
5	This option adjusts the screen refresh rate, [0..500ms].
6	This option adjusts whether run1 and run2 are played together or separately.
7	Toggles trajectory 1/2 on/off.
8	Play/pause trajectory 1/2.
9	Stop the playback of trajectory 1/2.

Table 5: Search Playback Options

Color and Highlighting

Explanations

Given the visual complexity, it is important to assist the user to quickly find what the user wants. Highlighting, to make certain things *clearly* stand out from the rest, is a very helpful visual aids to address this issue. With proper highlighting, the user can recognize important things immediately while minimizing information overload.

Highlighting is possible because human has pre-attentive visual processing system [7] in which certain visual features are distinguished by human visual system before conscious attention. Such

⁷This is do-able since VIZ is an offline visualization tool. Therefore, this playback speed does not reflect the true local search algorithm speed.

visual features that can be pre-attentively processed are: mixtures of colors/grey scale levels, objects' sizes/shapes/textures/orientations/labels, etc.

Understanding such pre-attentive process is the key to designing elements of displays that must be rapidly attended to. Making something significantly different from its surroundings on one of the pre-attentive dimensions ensures that it can be detected by a viewer without effort and at high speed.

However, different form of highlights have different performance in different context. Changing object's color is harder to differentiate compared to changing object's size when the number of different colors being used are high. However, if there are many different objects and we only use small number of colors, then changing object's size is more confusing and changing object's color is more preferred. In VIZ, we mainly used color⁸ as the major visual feature for highlighting important information.

Other than for pure highlighting, color is also good for coding/labeling/categorizing. If we are using the same color for different but logically related objects (even when the objects are drawn in separate visualization windows), human visual system will process them as being associated. Other visual features are not as good as color for coding information.

However, misuse of color can be actually distracting. Using too many different colors in a visualization system is actually confusing. Using color gradient/greyscale is also not always helpful as they are not good to code information [7].

After understanding the power of color and the needs for rapid attention to highlight important parts, we used color and highlighting in various parts of VIZ as follows:

1. In ST, OV, and FDC visualization, color is used to form contour map which quickly highlight the quality of regions in the search space. Changing the range of values in the ST control panel screen enables us to quickly differentiate solution qualities.
2. In ST visualization, the 'X' sign at the beginning of search playback quickly identifies the position (distance w.r.t best found) and quality of the initial solution.
3. In ST visualization again, the tip of the trail is drawn either as a 'cross' or a 'circle' to quickly identifies uphill and downhill moves. When re-played back slowly, one can quickly digest the up/down pattern in the solution quality on top of the movement information as drawn on the screen.
4. Still in ST visualization, we provided the capability to do mouse hover queries. Hovering your mouse on top of an anchor point in the ST visualization will reveal detailed information of that particular AP in the TBIC and PS visualization.
5. The cross sign in the FDC visualization is used to quickly indicate the position of the current 'delta Fitness' and 'Distance' of current solution w.r.t to best found.
6. In EB visualization, we draw various stripes of different colors to highlight different generic events that are found throughout the search playback.
7. The small red rectangle beside the playback toolbar to indicate which Trail 1/2 is active at the moment.

Explanation about default color selection:

1. We select the default color for search trajectory 1 as Red, search trajectory 2 as blue because they are the primary colors which are hardwired in human brains. Blue and Red are visually distinguishable: blue is usually perceived as receding and red as coming forward. This make them visually different.
2. For the contour map for the labeling of objective value/fitness, we adopt the following default color scheme: sea blue (good quality), green (medium quality), yellow (bad quality). These colors are the commonly used color to represent the height levels in common geography maps.
3. For the color of other GUI parts, we follow the following simple guideline: bold color for emphasis/attention grabber, calming color for normal usage.

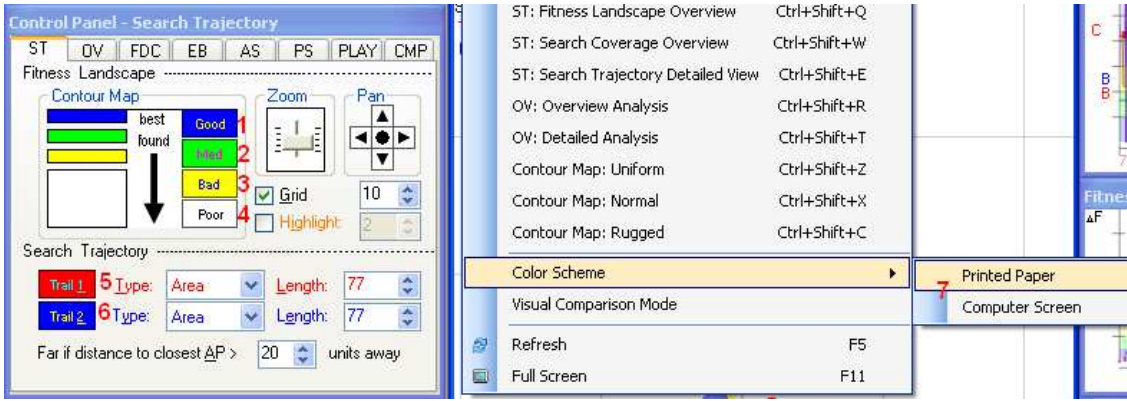


Figure 7: Color Scheme Options.

Items	Description
1,2,3,4	User can choose the color for the contour map.
5,6	User must be able to pick their own favorite color for search trajectory 1/2.
7	Some pre-set color schemes.

Table 6: Color Scheme Options

Color Scheme Options

Multiple Level of Details

Explanations

Figure 8: Level of Details, OV-overview vs detail, FDC-overview vs normal

Information Visualization (Visual Information-Seeking) Mantra:

Overview First, Zoom and Filter, then Details on Demand... [?].

Level of Details (LoD) technique: draw more (give details/granularity, highlight important ones) when zooming in and draw less (summary, hide non-important ones) while zooming out (overview). This is to ensure the limit of human brain is observed, as human cannot do micro manage, but only macro manage... The idea of zooming and panning is used in an interactive 3D map: Google Earth [3]. Btw, draw using MS Paint, a prototype screen of overview-focus box style... include the picture here...

1. Btw, now LoD is not for ST visualization only... Actually I've embedded it into OV (overview/detail), FDC (overview/normal)... The scaling feature that we have for OV chart also make that OV visualization have LoD capability...
2. Note that if you scale your visualization differently, you may probably infer different things, this is the power of level of details. Currently I use the maximum window size to scale each visualization screen...
3. If we want to do LoD in ST visualization, we must give option for auto-tracking. This will enable the user to track the current solution within the viewing window in the zoomed-in view... This is how we should do it: 'pause' for a while, scroll, and then resume search... to avoid flickers... However, I may want to just use overview mode for some time...
4. Larger display is obviously better, can we convince the user that the important information can be shown by VIZ using standard screen size?
5. Draw two search trajectory windows simultaneously, one overview, one zoomed-in/close-up (Google Earth use this kind of double visualization actually), this will be useful to argue that doing this simultaneously gives the user necessary overview (I know where I am) plus details (I know what are nearby). Later, if possible, how if we mimic some of Google Earth's features for VIZ?

⁸For further in-depth discussion about color, please refer to [?, 7]

6. When we zoom in, we can do more level of details... The problem is that when I zoom into the search trajectory visualization, I still don't know what kind of additional details to draw that can help understanding the search. We already log a lot of things, we can just present the data in LoD manner... Think... what should be drawn more when we do zoom in??. Note that human can only do macro manage, but if we zoom-in close enough, we must still reveal more data, draw more stuffs... I'm thinking why don't we draw distance information from current solution to those AP within vicinity, highlights the information about those close APs... perhaps by doing that we can infer more?

Text-Based Information Center (TBIC)

Some things are best displayed as text. However, adding too much text in the visualization will clutter the screen. Therefore, some details are displayed as text outside the visualization window. This is where TBIC window is mainly used for.

Similar to the pages in the Control Panel window, the content displayed in this TBIC is context-sensitive, according to which visualization window is being activated by the user. So, the textual information about a visualization is immediately shown as needed.

Other than additional textual information, we also compute some standard statistics which may be difficult to be seen in the visualization itself. Statistical summaries are objective, but they can hide some information, e.g. outliers or discontinuities in data. In VIZ, we are combining information visualization (which hypothesis must be verified/extracted solely by human visual system) and data mining (which the analysis are computed by computer via some statistical rules).

By default, textual information is also compared between Run 1 and Run 2 whenever possible.

Textual information that is available in TBIC:

1. A summary and visual data file information to inform the user of basic information.
2. When OV visualization is activated, a statistical summary about objective value: best found, average, etc, are displayed.
3. When FDC visualization is activated, a statistical summary of FDC: the r_{FD} coefficients, the average distance w.r.t best found, etc, are displayed.
4. When the search playback option is activated, a Run Time projection estimation for longer runs is displayed. This information is projected from the actual search time information in the RunLogs... This is to help the algorithm designer in planning the execution time for his/her algorithm.
5. There are detailed information of current anchor points, current solutions of search trajectory 1/2, and the tag information. These information are updated in concert with the appropriate visualizations.

Bibliography

- [1] M.R. Endsley. *Theoretical Underpinnings of Situation Awareness: A Critical Review*, chapter -. Endsley and Garland (eds). Situation Awareness Analysis and Measurement. Lawrence Erlbaum Associates, Mahwah, NJ., 2000.
- [2] S. Few. *Information Dashboard Design*. O'Reilly, 2006.
- [3] Google. Google Earth, <http://earth.google.com/>, 2005.
- [4] E. Tufte. *The Visual Display of Quantitative Information*. Graphic Press, 1983.
- [5] E. Tufte. *Envisioning Information*. Graphic Press, 1990.
- [6] E. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphic Press, 1997.
- [7] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, second edition, 2004.