Communicating Visually

Visual Communication is not a Natural Skill

As we’ve indicated in previous chapters, we humans are natural born storytellers; having amassed the essence of basic verbal storytelling by the time we’re barely 36 months old.

While we may be born with a basic ability to communicate audibly, our collective inability to master visual communication should not be a huge revelation to anyone who has lived through the Geocities and MySpace eras of the Internet. Yet, many of those who would shudder at the thought of bringing back the <blink> tag to our web browsers have virtually no issues schlepping a column or two of data into an Excel spreadsheet and walking away with a default chart image that can be cut and pasted into a PowerPoint document for an upcoming presentation. What causes this dichotomy between the acts of creation and perception and what can we do to fill in the gap?

To answer those questions, we need to understand how we create and receive images, a process called *“visual literacy”*, which has been defined by John Debes [ref] as:

*“…[the] group of vision-competencies a human being can develop by seeing and at the same time having and integrating other sensory experiences. The development of these competencies is fundamental to normal human learning. When developed, they enable a visually literate person to discriminate and interpret the visible actions, objects, symbols, natural or man-made, that he encounters in his environment. Through the creative use of these competencies, he is able to communicate with others. Through the appreciative use of these competencies, he is able to comprehend and enjoy the masterworks of visual communication.”*

So, visual comprehension is not a passive act but a very deliberate one, with our eyes taking in images and our brains interpreting, processing and deriving meaning from them—a process also known as *decoding*. While humans may have wrapped a definition around this process in the 20th century, this is old news…approximately 60,000 years old (give or take a century).



Figure Cave painting (left); pictograph (right) [source ref]

In some ways, it’s no surprise that one of our first acts as we emerge into this world is to exercise our vocal cords and make our presence known to everyone in earshot. After all, we spend nine months being subjected to aural inputs with no opportunity to respond in-kind. But, what is the spark that ignites the need to communicate with images?

Cave paintings (figure 1) remain to this day sole artifacts of one of the first forays into mass, image-oriented visual communication. While it’s impossible for our modern minds to derive definitive meaning from these images, they are examples of *visual literacy* in action. The aforementioned spark occurred in the minds of a scant few prehistoric PowerPoint creators that both drove *and enabled them* to transcribe items from their three dimensional world into two dimensions with as much precision as implements at that time would allow. It’s unlikely these images stood on their own merit, and one can picture a tribal shaman—the regional sales rep of the day, as it were—animating painted scenes with his hands to help the audience understand what he as trying to convey (which was, most likely, identifying the animals were good to eat and the ones that should be avoided, which definitely has more practical benefit than most sales presentations today do).

Fast-forward 50,000 years or so to when the first petroglyphs (figure 1) were produced and we see a creative evolution occurring which produced images that are more intricate and complex, demonstrating that the visual literacy of both the senders—still limited in number—and receivers increased significantly. The widespread discovery of similar-styled petroglyphs across nearly every continent provides further evidence of the aforementioned collective need to communicate and be understood visually. Yet, without shaman-provided context, it’s likely these standalone visualizations still would not communicate very well outside a small, close-knit group.

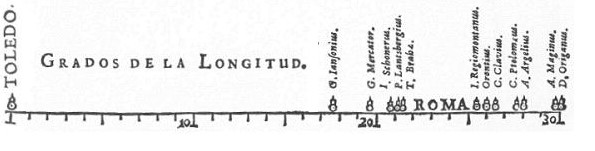


Figure 1644 chart of estimates of distances between Toledo & Rome from various navigator sources at that time by Michael Flortent van Langren [ref]

As we sail up through the timeline into the 17th century, picture the scene in King Phillip IV’s Spanish court after news of yet-another shipwreck found its way to his ear. Then, pan left to see the wheels turning in the mind of a middle-aged court mathematician as he considers a scientific approach to solving this crisis of commerce.

The results of this, albeit highly contrived, scene would be a dramatic increase in the *visual literacy* of humans of that era due to the creation of what would now be considered a simple one dimensional line graph by Michael Florent van Langren. Van Langren fancied himself a “sphereographer” and set out to find a way to show the court that these shipwrecks were the results of incorrect assumptions of longitude distances from Toledo to Rome. This exercise also provided the opportunity for van Langren to display his longitudinal prowess.

He fused visualization concepts from peers and predecessors such as Nicole Oresme, Albert of Saxony, Leonardo da Vinci, Nicholas of Cusa and, no doubt, many others and relied on the fact that his audience was also somewhat familiar with these teachings. The result is the chart in figure 2, which is considered the first known graph of statistical data.

While van Langren’s chart is good, it didn’t just “appear”. It took real effort to piece together how one could succinctly and effectively communicate the problem with the data and ultimately ended up relying on the receiver’s ability to decode a new communication method. If only the 17th century had a github equivalent for us to be able to peek at the iterations that did not survive the test of successful communication.

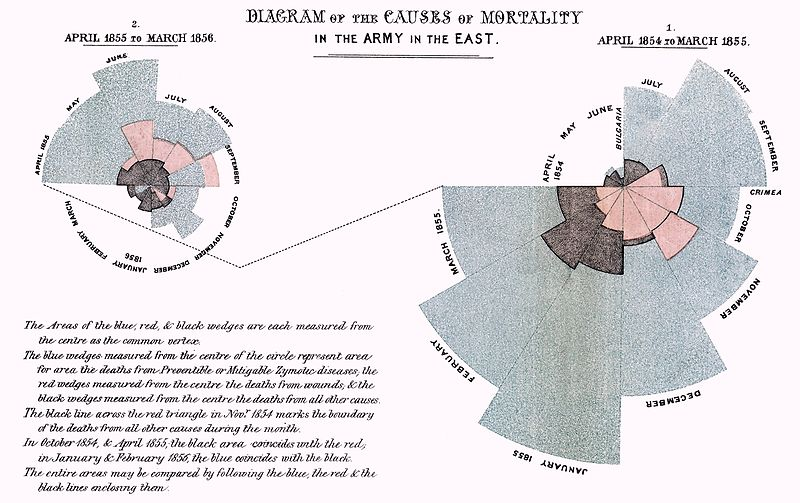


Figure Florence Nightingale's "rose diagram" of patient mortality rates

Marching, now, into Crimea during the mid-1800’s we find British soldiers dying by the thousands and also see the medical profession charging to the take the lead when it comes to pushing our decoding boundaries to help communicate both complex and critical analyses.

Florence Nightingale was a highly capable nurse assigned to a military hospital where sick and wounded British soldiers were sent (ostensibly to get better). She was also a very capable statistician and, as it turns out, visualization pioneer. Her domain expertise combined with her statistical abilities led her to the discovery that the majority of soldiers were dying not of battle, but through diseases picked up in both the army camps **and** army hospitals. The question remained as to how to most effectively communicate this discovery to garner a call to action; a task we face in IT weekly, if not daily.

Her data set was pretty basic: time-series categorical data with counts that could have simply been presented by bar or line graphs—both easily decoded charts by that time—or even by a simple set of tables. Yet, the goal of these diagrams was not merely to reproduce the underlying data with precision, but also to *visually connect with the receiver* and show trends and interdependencies in a compelling way.

The result was her now famous “Nightingale rose diagram” (figure 3) which first appeared in the *Diagram of the Causes of Mortality in the Army in the East*. Such a diagram required inspiration on the part of creator and relied upon the receivers’ ability to extend their comprehension of other known charts in order to fully grasp the severity of the hygiene problem. Her efforts were a success but yielded limited wartime impact as the conflict ended within a year of the Sanitary Commission taking action to improve conditions for soldiers.

Since these early successes, we have had a wealth of opportunity to both enhance our encoding capabilities and investigate the science behind how we go about decoding images. The dawn of the 20th century brought with it many psychological and medical discoveries that have enabled visual communicators to move from mere inspired trial-and-error to understanding how we see and process images.

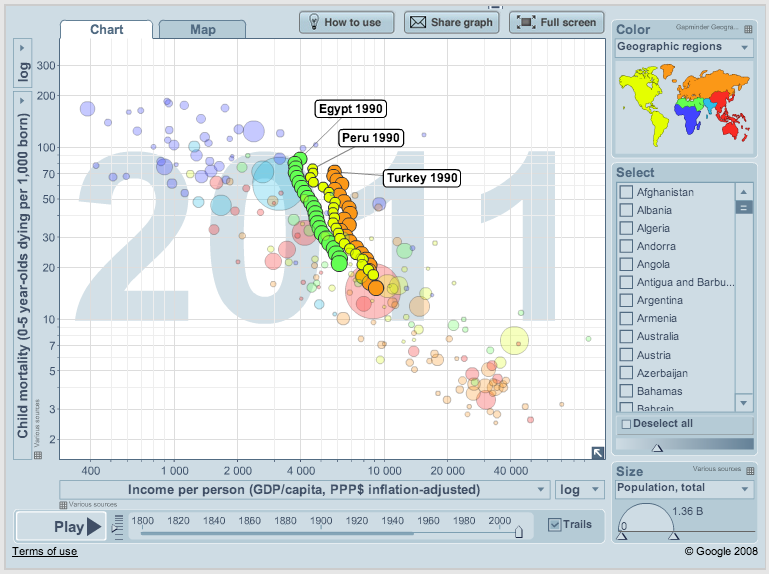


Figure Gapminder interactive/animated view of child mortality rates by country through time

A modern example that builds upon this work comes from Gapminder [ref], a non-profit foundation that promotes sustainable global development. A key component of their mission is fulfilled through the use of dynamic visualizations to “*…[fight] devastating ignorance with fact-based worldviews everyone can understand.”*  The tools that help produce these data-infused visual stories were designed with the knowledge of how we best decode these images, providing a pre-configured canvas to aid even the most nascent communicator.

Gapminder incorporates color, animation and directed exploration controls to help users navigate through extensive data sets to and gain insights into complex topics such as child mortality rates (figure 4). By letting the audience control the exploration, Gapminder offers the potential to draw new discoveries or identify possible parallels that would not be possible through static charts. Users are then able to share their insights with a simple “share graph” button. This simple feature may be the most powerful one of the tool since data visualization is difficult for many people, yet they desperately want to be able to communicate visually, especially after they’ve been the recipients of effective visual communication.

It should be pretty clear that effective visual communication takes work on both the part of the senders and receivers. Even with well-designed toolsets, it’s vital that we understand how the image-to-understanding decoding process works, especially if we wish to bring multiple elements together or become 21st century visual literacy pioneers.

Cognitive Science: Decoding the Decoding Process

Understanding how an image will be decoded may be the most fundamental component of visual communication. You wouldn’t think of writing an e-mail in Spanish if the recipient could only read German because you know they could not decode the text. How, then, can we expect to communicate through visualizations without first knowing the visual decoding process? To understand this process, we need to look (briefly) into a field of study called “cognitive science”.

That term may sound fairly daunting, but it can be defined simply as: the bringing together of concepts from philosophy, psychology, artificial intelligence, neuroscience, linguistics and anthropology to help figure out of the mind works [ref]. In many ways, it’s a form of detective work—piecing together clues from many sources to solve a mystery.

Cognitive scientists really want to know how we humans think, and they posit that how we think is best defined by

* how our minds represent things, and
* the operations our minds perform on those representations.

When it comes to visual communication, we need to solve the mystery of how *images* are decoded so we can figure out how what we create will be received. To do that, we need to understand biological and neurological aspects of ocular image processing.

One clue that biology detectives have found to help solve this mystery is that it takes the human eye approximately 1/20th of a second to glean the meaning of a complex visual scene (this is one reason there are twenty-four frames per second in a movie frame). Even though we will undoubtedly have more time than that to ponder the content of an image, much of that extended reflection will be based *and biased* on that initial information retrieval. Our choices of color, brightness and symbols will be processed in less time than it takes our eyes to blink, so we really need to make sure we do our best to encode our visualizations properly or we risk having them be misinterpreted.

To help make some of the theoretical concepts more concrete, the next section will frame each core topic with a practical visualization challenge and then give you the information you need to make the best decision.

Signal Detection and Magnitude Estimation

**Visualization challenge**: *You’ve been asked to put together chart on the number of critical, high, medium, low and very low vulnerabilities on your Windows servers. How do you best encode this for a printed report? The answer is not as obvious as you might expect.*

If you whisper to someone sitting next to you in the library, there’s a fairly good chance they’ll be able to hear you and understand what you’re saying. However, if you try whispering to someone in the middle of a rock concert, your signal may have severe difficulty of getting through the noise.

Visual communication has similar issues to that aural example, with the signal being brightness and color broadcasts to eyes versus sound waves to ears. Because we continuously process visual stimuli it may seem that w inherently already know all there is about this fundamental concept, but perhaps the following simple example will help show that further research is in order.

Take a look at figure 5, below:

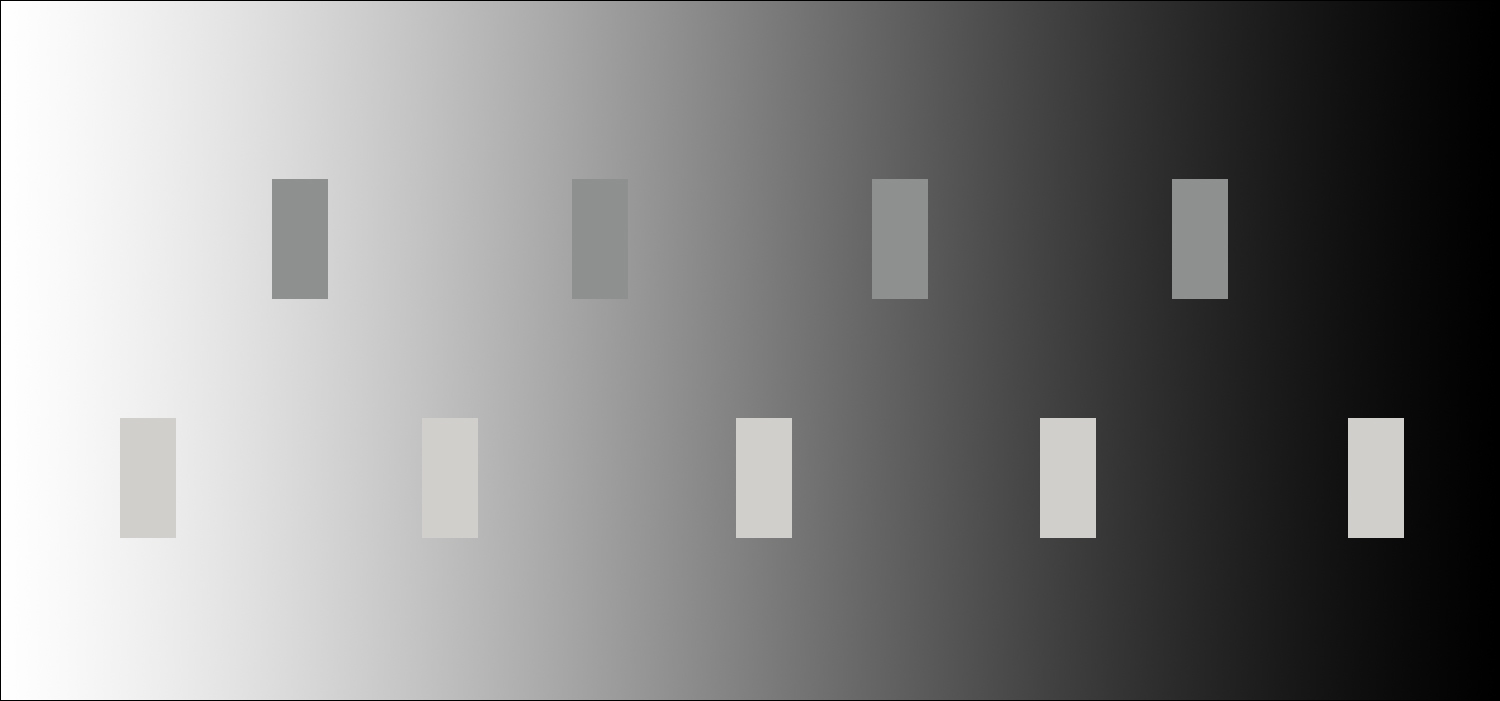


Figure Visual signal and noise detection illusion

You should see two rows of rectangles that appear to be filled with different levels of gray. What you are actually seeing is an artifact of the decoding process **since the top row of rectangles are all the same shade and the bottom row of rectangles are also all the same shade** (albeit, a lighter one). You can validate this with your favorite color picker by looking at the source for that graphic in this chapter’s section on the web site. The background gradient is the surrounding “noise” and the elements of the rows of rectangles are the “signals”. This gentle reminder that our innate assumptions about what the receiver “should” interpret can often be wrong is also a good introduction to the principles of visual signal detection.

Weber’s Law – Spot the Difference

If we’re going to use differences in brightness (or shade) to communicate, it would be helpful to know just how different we need to make these shades so that the receivers will notice. But, how can we possibly know what those difference values are? The most straightforward way would be to layout various examples of these brightness differences in front of our coworkers and record the results of their reactions to various combinations. That sounds like—well—*work*, and most of us have full schedules already.

Thankfully for us, Dr. Ernst Heinrich Weber conducted numerous empirical studies to attempt to determine the relationship between a various kinds of physical stimulus and the perception of the intensity of said stimulus. This test was performed across many senses—including vision—and culminated in the principle of *just noticeable difference* (JND) [ref], or the smallest detectable difference between two levels. For normal human eyesight under optimal conditions there are approximately 1,000 JND steps. However, when our eyes are required to adapt to different lighting conditions (think disparate monitor calibrations, paper brightness, full sunlight vs dark room) the number of steps reduces to approximately 200.

As the brightness illusion demonstrated, an image’s environment also makes a huge difference to perception. When there are many surrounding intensities the primary signal must be bright enough to overcome the processing in that post-reception step. If we take the previous example and crank up the brightness on all the rectangles to full white (figure 6) our eyes have a much easier time separating the signal from the surrounding noise.

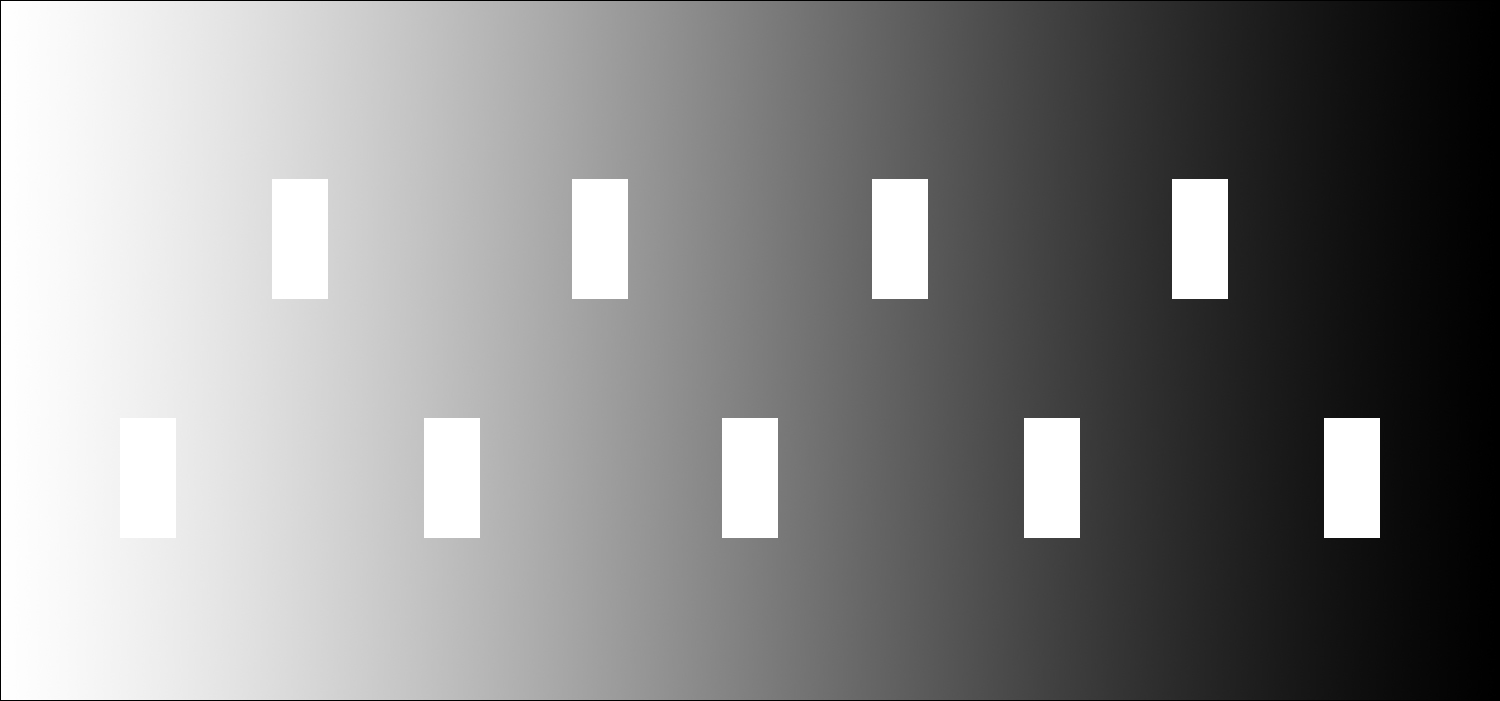


Figure Increasing the intensity of the signal helps cut through the noise

Understanding Weber’s law can help us make better decisions when developing our visualizations. Because we only have the ability to detect a fixed number of steps and that our minds seem to have an inherent concept of order when it comes to brightness (i.e. “A is brighter/darker than B”), **brightness variations should be used to** **encode ordinal variables** and we should strive to keep the number of encodings small and have the magnitude between different brightness levels as large as possible. For our visualization challenge, the shading choices in Figure 7 would be much better than choosing the color defaults in Excel and just hoping they work out well in print.



Figure Good, distinct choices for C/H/M/L/VL

criticality encoding in black & white

Stevens’ Law — Encoding Magnitude

**Visualization challenge**: *You did such a good job in your previous report that the Windows manager would like your help tracking and reporting the number of Windows critical vulnerabilities over the course of the next four weeks as the systems get patched. What encoding will most accurately convey the magnitudes across the weeks?*

Dr. Stanley Smith Stevens was also interested in determining the relationship between magnitude of physical stimuli and the way humans perceive the strength/intensity [ref]. He incorporated far more continuums than Weber did in his trials, and—again, thankfully for us—he included tests for visual length and area. (As an aside, it’s interesting to note that both Stevens and Weber managed to acquire test subjects willing to be subjected to *electric shock* and other forms of real pain for their studies, perhaps making them predecessors to our modern day Mythbusters.)

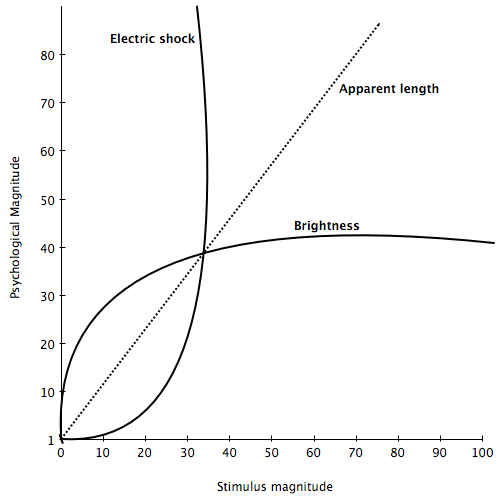


Figure Reproduction of the results of Stevens' stimulus vs

psychological response results [ref]

Figure 8 is a reproduction of one of Stevens’ graphs found in *The Psychophysics of Sensory Function* [ref] explaining the results of his studies. Length interpretation is a perfect diagonal line, meaning that we gauge the stimulus intensity correctly when comparing between different lengths. It also shows that we over-estimate electric shock impulses and under-estimate brightness changes. This means we **are far better off using length to encode magnitude comparisons** than we are delivering a proportionally good shock to the receiver (bummer) or using brightness.

Other charts from Stevens show that circular area determination falls just above the brightness curve, meaning that **receivers tend to underestimate the values when comparing objects by area**. If circular area is chosen for the encoding, the sizes should have larger, disproportionate scaling vs absolute scaling given our tendency to underestimate the comparisons.

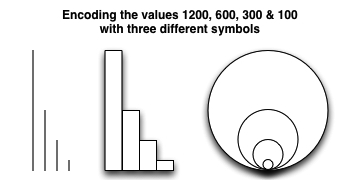


Figure Possible acceptable ways of conveying the decrease in critical

vulnerabilities from 1200 to 600, 300 and 100 over four weeks

For our visualization challenge, look at figure 9 and consider which comparison encoding you decoded faster. Unless you’re an interstellar visitor, you should have noticed your decoding process was almost instantaneous for the lines and bars while it took you some extra processing time to make the same comparison with the circles, *even with the proper scaling choice for the circular chart*. We may think circles are pretty (and they are), but they don’t help us make accurate decisions faster.

Comparing and Ranking Elementary Perceptual Tasks

**Visualization challenge**: *The UNIX team has gotten a bit jealous of all the help you’ve been giving the Windows team and they’ve asked you to help them produce a chart that shows the number of critical vulnerabilities across all the UNIX systems broken down by business unit (SBU). Their goal is to try to increase the length of change windows for system patching. Before you reach for that pie chart (you know you want to) let’s see what other choices might be out there to help the UNIX team accomplish their goal.*

The act of building a chart or graph involves the encoding quantitative data—*numbers*—(vulnerability counts in our challenge) and categorical information—the *descriptive labels* that tell us what those numbers measure—(SBU UNIX systems in our challenge) using a selection of position, shape, size, symbols and color. The choices we make will either aid or detract from the ability of the receiver to accurately decode the image. If only there were a reference for how we humans perceive the visual presentation of quantitative information…

Cleveland and McGill — Decoding Accuracy

Unlike van Langren (who had to actually *invent a line chart…#respect*) **you** are a dweller of the 21st century and can reference a study [ref] that William S. Cleveland and Robert McGill performed back at the tail end of the 20th century for help. Cleveland and McGill set out to rank human elementary graphical-perception tasks from ten encoding types, which ultimately group into an ordered list of seven (figure 10), and support and extend the findings in Stevens and Weber’s studies.

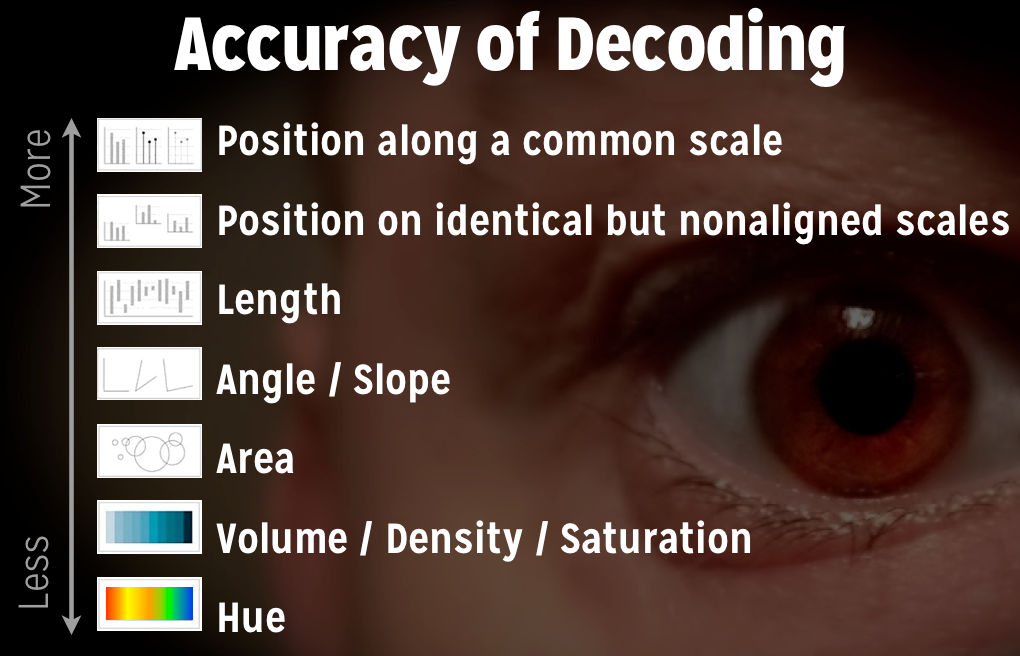


Figure Cleveland & McGill decoding rankings [ref] NOT FINAL IMAGE EXAMPLE ONLY

We decode position and length—bar and dot charts—with more accuracy and more quickly (as was seen previously) than angles and slope—pie chars. If we truly want to help the UNIX team show that SBUs A, B and D need to provide more time during change windows, bar charts are clearly the way to go (figure 11).

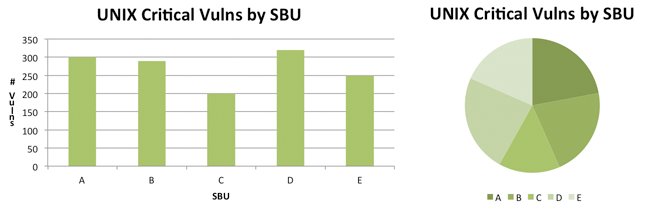
****

Figure Bar charts communicate more quickly and accurately than pie charts

Mackinlay — Data Drives Visualization Choices

Sadly, pie charts are not so easily dismissed. Jock Mackinlay, formerly a professor at Stanford University and now a VP at Tableau Software, has worked [ref] on designing a algorithms that could automatically generate appropriate charts and graphs depending on the types of data being encoded. His work extended the efforts of Cleveland, McGill, Stevens and Weber by taking into consideration the data types when building the rankings.

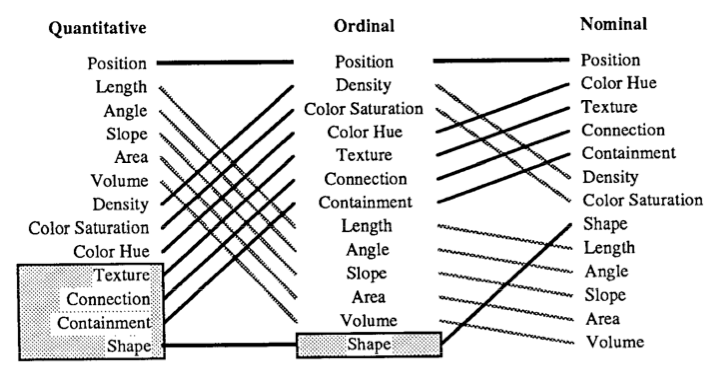


Figure Mackinlay's ranking of perceptual tasks (NOT FINAL CHART) [ref]

Mackinlay’s research showed that his predecessors’ findings held true for *quantitative* data (i.e. data that has a precise numerical value) but not necessarily for *ordinal* data (i.e. data with an inherent order relationship, such as [high, medium, low] vulnerability criticality) or *nominal* data (i.e. data that represents members of a certain class such as [Windows, UNIX, mainframe] servers). Figure 12 shows these modified rankings along with the movement of the individual encodings across rankings. If you look at them closely, Mackinlay’s idea to focus on the data type makes quite a bit of sense.

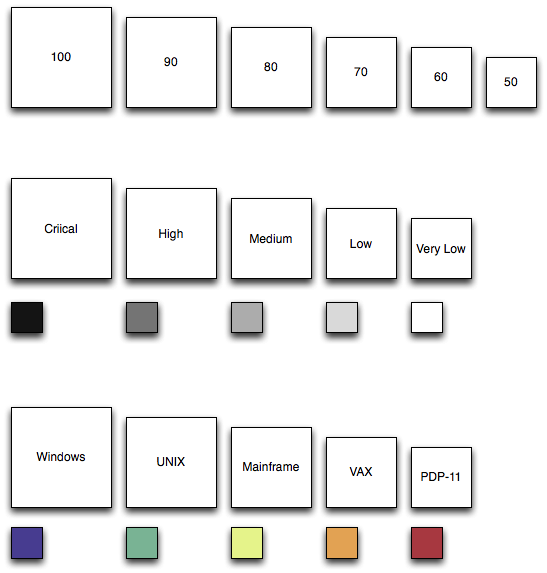


Figure Examples of encoding with Mackinlay's data-based rankings

In figure 13, area does a decent job in the top row where we are comparing quantitative vulnerability counts, especially since we used squares which gives the receiver an inherent length to work with (this is why bar charts are so cool). Area can work for the *ordinal* vulnerability severity rankings in the middle, but shifting the black density enables the same encoding without the extra processing overhead of area comparison. For the *nominal* system type variables on the bottom the area changes do differentiate the types, but cause our brains to create an ordering where there isn’t one. Color hue achieves the desired goal without implying order or distracting us with area comparisons.

Encoding Multiple Attributes

**Visualization Challenge:** *You’ve been asked to give a quick overview of the number of vulnerabilities across all system types by business unit (SBU). However, you’ve been told that there’s only room/time for one more slide. How can you best paint this picture with just one chart?*

The importance of the concepts presented so far become even more pronounced when faced with the challenge of having to use multiple encoding techniques within one chart. Color, shape, size, brightness, slope, etc. will either facilitate communication and understanding or introduce confusion. It’s possible (and common) to come up with a chart that represents the data well, but does not convey the message its creator wanted (think back to the choices Florence Nightingale had when she chose to create a whole new visualization type). So, you need to understand what message you’re trying to convey and what encoding techniques will communicate that story most effectively.

The charts in figure 14 are based on the following data table collected by our fictional vulnerability and asset management system:

Table Missing patch counts by SBU & OS-type

|  |  |  |
| --- | --- | --- |
| SBU | Patch OS | Missing Critical Patch Count |
| A | UNIX | 50 |
| A | Windows | 100 |
| A | Mainframe | 1 |
| B | UNIX | 10 |
| B | Windows | 1 |
| B | Mainframe | 0 |
| C | UNIX | 30 |
| C | Windows | 20 |
| C | Mainframe | 1 |
| D | UNIX | 40 |
| D | Windows | 120 |
| D | Mainframe | 2 |
| E | UNIX | 75 |
| E | Windows | 1 |
| E | Mainframe | 0 |

How can we best encode this information to show how well each SBU is keeping up with critical patches for each of the platforms they use?

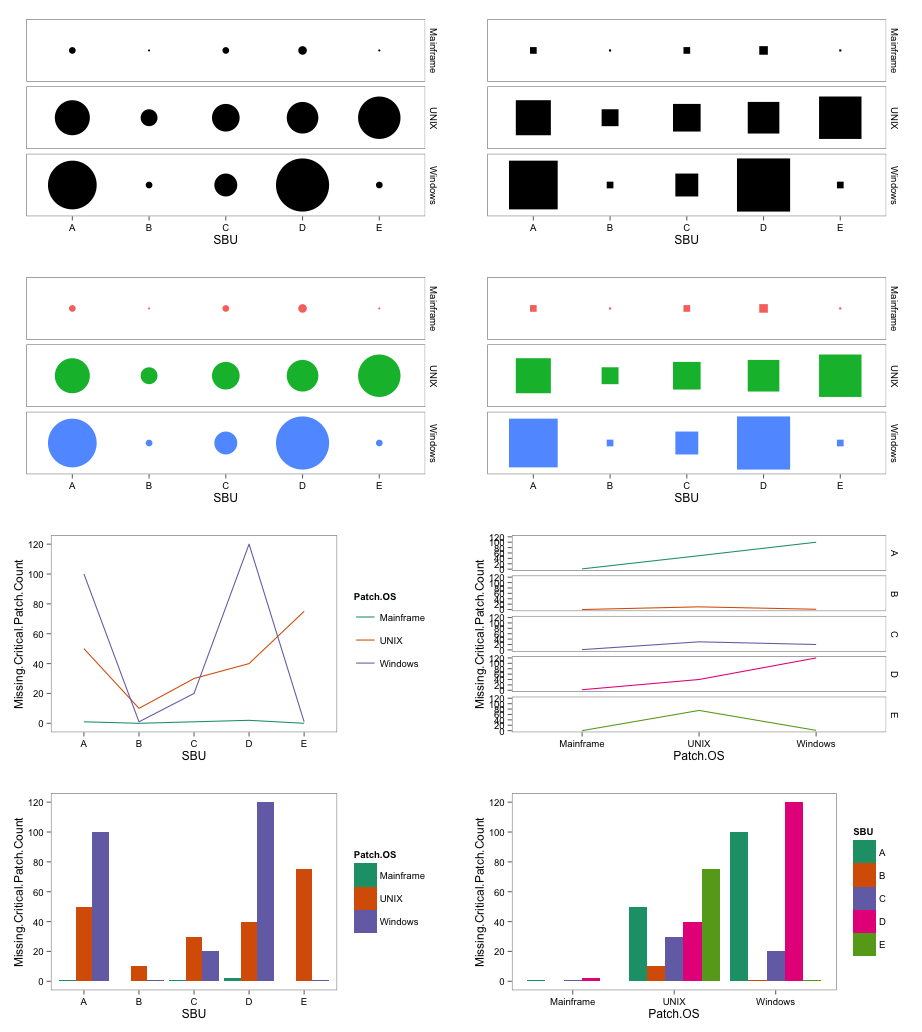


Figure Multiple encoding techniques NOT FINAL PRODUCT; NEEDS MORE/BETTER LABELS

The first attempt (row 1) tries to use position and area to encode the data, but you should notice that your eyes keep bouncing from shape to shape as your mind tries to calculate area and hold that data as you move between elements (we’ll cover more on the concept of visual memory at the end of this chapter). Most of us need to workout more, but not necessarily our eye muscles and synapses.

NOTE: *a single strip* bubble chart with a small number of elements *can* be an effective communication tool, but remember this example when you get an idea to go crazy with bubbles.

Adding some encoding redundancy to the system type with color to the same charts (row 2) makes it a bit easier to at least see SBU A and D are not exactly keeping their systems patched well, but it still requires some effort to keep track of the information and doesn’t paint the complete picture well. It would also add confusion if translated to black and white as the introduced gray levels would hint that there is an implied order where none exists.

The line charts (row 3) do a fantastic job of tricking the user into thinking that there is some continuity between the points when there is none for this data set. In the chart on the left, it almost looks like SBU C *caused* the poor Windows patching performance in SBU D. It should also help show that sloped lines are dangerous tools in the wrong hands, especially when axes scales are skewed.

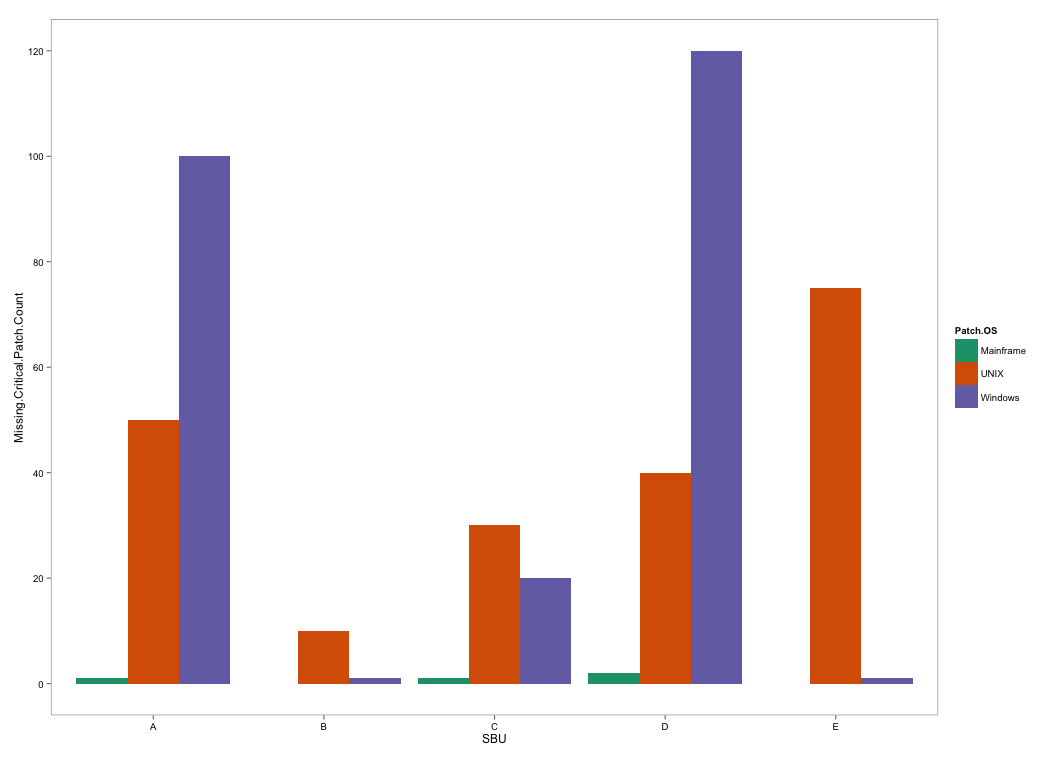


Figure Bar charts FTW! NOT FINAL PRODUCT

Finally, the humble bar chart comes through again (row 4), with the one on the left (and in figure 15) very clearly showing the security patch performance of each SBU across all platforms. Color makes the nominal data (system type) distinct while the bar lengths accurately show the quantitative data (missing patch counts) and bar positions help with the proper groupings.

Even though you are now armed with the reference tools from multiple empirical studies, nothing can truly compare with testing out visualizations and looking across a set of examples to see what is or isn’t working (like we just did in figure 14). Tools like Tableau or R make this very easy to do. The R code that generated figure 14 is below and available on the website/github repository for the book.

library(ggplot2)

library(ggthemes)

library(grid)

library(gridExtra)

library(RColorBrewer)

*# read in the data file of missing patches*

df = read.csv("patch.csv")

*# setup common plot parameters*

base = ggplot(data=df) + scale\_fill\_brewer(palette="Dark2") + scale\_color\_brewer(palette="Dark2") + theme\_few()

*# setup row 1 bubble and square plots*

bubbles = base + geom\_point(aes(x=SBU, y=1, size=Missing.Critical.Patch.Count, fill="black")) + facet\_grid(Patch.OS~.) + scale\_area(range=c(1,25)) + theme(legend.position = "none", axis.text.y = element\_blank(), axis.title.y = element\_blank(), axis.ticks.y = element\_blank())

squares = base + geom\_point(aes(x=SBU, y=1, size=Missing.Critical.Patch.Count, fill="black"), shape=15) +facet\_grid(Patch.OS~.) + scale\_area(range=c(1,25)) + theme(legend.position = "none", axis.text.y = element\_blank(), axis.title.y = element\_blank(), axis.ticks.y = element\_blank())

*# setup row 2 colored bubble and square plots*

colored.bubbles = base + geom\_point(aes(x=SBU, y=1, size=Missing.Critical.Patch.Count, color=Patch.OS)) + facet\_grid(Patch.OS~.) + scale\_area(range=c(1,25)) + theme(legend.position = "none", axis.text.y = element\_blank(), axis.title.y = element\_blank(), .ticks.y = element\_blank())

colored.squares = base + geom\_point(aes(x=SBU,y=1, size=Missing.Critical.Patch.Count, color=Patch.OS),shape=15) + facet\_grid(Patch.OS~.) + scale\_area(range=c(1,25)) + theme(legend.position = "none", axis.text.y = element\_blank(), axis.title.y = element\_blank(), axis.ticks.y = element\_blank())

*# setup row 3 line charts*

sbu.lines = base + geom\_line(aes(x=SBU, y=Missing.Critical.Patch.Count, group=Patch.OS, color=Patch.OS))

os.lines = base + geom\_line(aes(x=Patch.OS, y=Missing.Critical.Patch.Count, group=SBU,color=SBU)) + facet\_grid(SBU~.) + theme(legend.position = "none")

*# setup row 4 bar charts*

sbu.bars = base + geom\_bar(aes(x=SBU, y=Missing.Critical.Patch.Count, fill=Patch.OS), position="dodge")

os.bars = base + geom\_bar(aes(x=Patch.OS, y=Missing.Critical.Patch.Count, fill=SBU),position="dodge")

# plot the charts

grid.arrange(bubbles,squares, colored.bubbles,colored.squares,

sbu.lines,os.lines, sbu.bars,os.bars,ncol=2)

Understanding Gestalt

Visual Processing