Assignment P2

Kougar Lott
klott3@gatech.edu

1 QUESTION ONE – TASKS IN AN HOUR

1.1 Tasks in an Hour at Work

To explore the range of tasks I do during a given hour, I chose an hour I spend at work outside of group activities such as meetings, standups, or trainings. I recorded five of these tasks below in Table 1. The first couple tasks involved me trying to manipulate a finite element analysis (FEA) model – an abstraction of a three-dimensional (3D) part – in a graphical user interface (GUI) called ANSA. The third task is one I do periodically, which is to check on simulations I have running our high-performance computing cluster (HPC) via a command line interface (CLI). The last two tasks involve me editing a shell script with a text editor then sending it to a coworker over instance messenger to help them do their own tasks on our HPC.

Table 1 – Tasks in an hour at work.

Task	**Goal**	**Interface**	**Object**
Orient a 3D part	See area on part to change the design of	Desktop GUI - ANSA	FEA model
Move a feature on a 3D part	Generate a new part for a variation study	Desktop GUI - ANSA	FEA model
Query HPC on simulation status	Analyze results of simulation	CLI - PUTTY	FEA Simulation
Send file to coworker	Help coworker run MATLAB on HPC	Desktop GUI – Microsoft Teams	Shell script
Edit a shell script	Help coworker run MATLAB on HPC	Text Editor - VS Code	Shell script

1.2 Directness and Invisibility of Tasks

1.2.1 *Task One – Orient a 3D FEA Model*

To accomplish the task of orienting a 3D model in ANSA, there is relatively close level of directness in interacting to the object itself – the 3D part. After importing the part file into the interface, the user can see the model on display while also zooming in or rotating the part in space with the mouse by clicking on it. A snapshot of the interface is provided on Figure 1.

Invisibility of interaction in this task was good as manipulating the object in space was almost instant (seconds). This can be attributed to the fact that most mouse controls with 3D software is similar: click and drag to rotate, middle mouse button to pan, scroll to zoom. However, those actions have only became invisible to me after using 3D software in industry for a few years. It used to be that I would instead click the buttons on the GUI for what specific orientation function I wanted to perform.

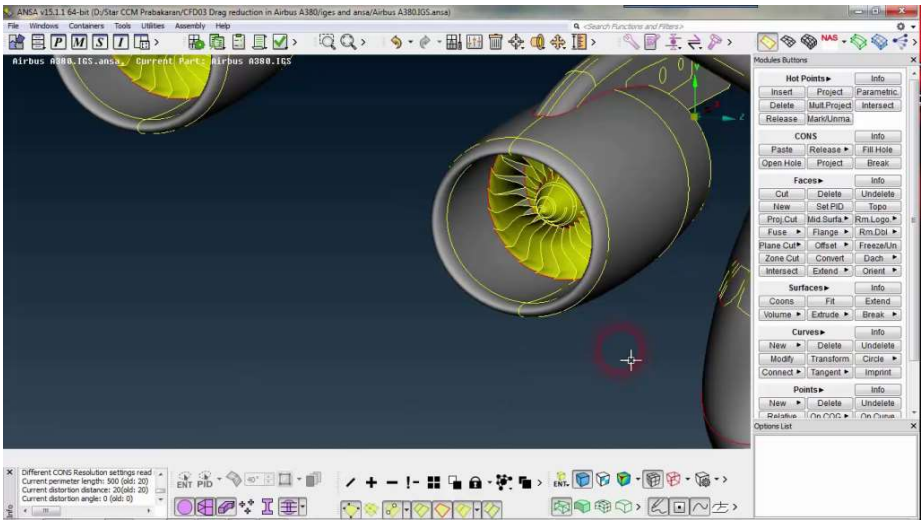

Figure 1 – An example screenshot of the ANSA GUI (Source: LearnCAEBlog).

1.2.2 *Task Two – Move a Feature on a 3D FEA Model*

To accomplish the task of moving a feature on the part I just oriented, the interaction felt a little further away. I first needed to identify the menu to allow me to move features, which felt more like looking something up in a dictionary. However, the menu itself was easy to use as the process mostly involved me again clicking the areas of the part I cared about then, then clicking and dragging them.

The invisibility of this task was not as good as selecting the feature-moving mode in the GUI took a few minutes. I first needed to click the search bar, type in a few keywords for what I wanted to do until the action I wanted appeared in the results, then an arrow dropped down which pointed to the button I was looking for. Despite this, the design is good via affordance (showing the user what to normally do). However, I used to not know about the search bar, in which I spent many minutes looking for the button I wanted: this is something that only became apparent in learning.

1.2.3 *Task Three – Query HPC on Simulation Status*

I often need to query our company's HPC to find out how a simulation I am running on it is doing (so I can either know when to analyze results of the simulation or debug something about it). The interaction generally feels far from the object (the simulation) as I needed to type some commands in a CLI, then get answers back as text such "running", "failed", "done", or "pending". In this case I am not really manipulating the object at all, just getting some info at a distance.

This task is mostly invisible to me: I spend maybe a second or two typing the command, or just pressing the up arrow and return to reuse a command. I would say this interface only became invisible due to learning. I used to spend a lot of time with this interface as I needed to look at manual pages or help documents to find the commands and options necessary to use our job manager.

1.2.4 *Task Four – Sending a File to a Coworker in Instant Messenger*

As part of my hour of recording tasks, I needed to send a shell script to a coworker so he could then get his MATLAB scripts running our HPC. Since he approached me on the instant messenger in Microsoft Teams, I tried sending the file over the messenger. The process felt mostly direct: I could just click and drag a file from my computer desktop to the chat window and it would send it.

Despite the directness, this process was not invisible at all – it took around a minute. Since the script had an ".sh" extension, the messenger ended up rejecting the file (*after* uploading!) due to security policy. I tried renaming to ".txt" which again failed, then ended up zipping the file which worked. In this case, I spent more time thinking about the interface due than the task, which was annoying – it would have been nice if the program could have then made suggestions on how to pass these kinds of files safely.

2 QUESTION TWO – COMMON AND INVISIBLE TASK

2.1 The Task and its Components

A common task I perform that has become invisible to me via learning is using my mountain bike to get around the local trails. The obvious components of the task that needed to be learned are balancing, steering, and putting power down through the pedals. However, these are common among all types of bike riding so I will focus instead on the elements specific to trail riding. These include modulating the front and rear brakes with their corresponding levers according to traction of the front and rear tires, leaning the bike over during sharp corners, dropping the seat with a button on the handlebar then leaning back on descents, and adjusting gears via a shifter on the handlebar before ascents.

2.2 My Thought Process Now and Why

For most aspects of using my bike's interface, I do not think about them anymore due to painful experiences of not following them in the right amount, order, or the right time.

For instance, I used to think about modulating my front and rear brakes depending on my speed, angle of descent, and tightness of corner (declarative knowledge passed on by experienced riders). While this knowledge gives enough direction to prevent serious injury, I usually did not discover the limits of my bike until after I either low-sided the bike with too much rear brake or flew off the trail with too much front brake. Today, I can feel the tell-tale signs through the seat and handlebars when the tires are starting to lose traction and know to release brake pressure automatically. This is a similar story to learning how much to lean the bike over during a corning event.

For learning how to drop the seat and lean back during steep descents, I used to try to survey the environment for how steep the trail was, frantically find the button to unlock the seat on my handlebars, push the seat down, then ultimately go flying over my handlebars anyways because I did it all to slowly or out of order. Today, after experiencing those mishaps, I subconsciously perform all these operations smoothly and order simply by feeling where my center of gravity is over the bike (and by moving my unlock button to a more convenient location near my hand).

For knowing which gear I need to be in during a steep climb, I used to do the process with some trial and error: I would simply drop a gear or two seeing a climb coming, hope I chose wisely to prevent shifting under load, coming to a stop if chose poorly, then needing the walk the bike up (or falling over if my feet were clipped into the pedals). Today, I can look at how steep a hill is and have an idea of how many gears I will need to shift to accommodate the change ahead of time.

2.3 How to Redesign the Interface

One of the best ways I can think of to help make the whole mountain bike interface more invisible faster is to provide real-time information about the bike to the user via augmented reality or a heads-up display (HUD). If the information is only contained on the bike itself, it is mostly useless as the user needs to pay attention to their surroundings. If info could be supplied to the user's vision, like Google Glass, they could know critical information before it is too late. Information like tire traction could give instant feedback to the user if actions they are doing are reaching the bikes limits without needing to find it via experience (a crash). Information like angle of the trail in front of the user could trigger warnings to the user to either drop their seat or how many gears to shift to have a chance of getting up a hill. Eventually, the user would probably ignore much of this information to suit their own riding style but would help them understand how to use their mountain bikes much quicker and less painfully.

3 QUESTION THREE – SENSES

3.1 Cooking a Meal – Human Perception Feedback

While cooking a meal, such a salmon fillet dinner, there are a variety of perceptive feedback signals involved to make it possible to complete (at least safely and edible). For this task, the user needs to bake the salmon itself, make a sauce to glaze the fillet with, sauté up some vegetables as a side, and cook rice to serve everything over – all of these subtasks involve some sort of perceptive cue to aid the user.

Visual perception is needed to identify if the food is visibly cooked via changes in color (browning of the salmon) or texture (wrinkling of vegetables). Auditory perception is involved as the cook needs to listen for timers or listen for sizzling

to check if ingredients are hot enough when sautéing. Haptic feedback is involved when testing if meats are cooked through such as feeling tenderness of the fillet itself or feeling stiffness of vegetables before cooking to either know which ones are not ripe enough or have gone bad.

3.2 Cooking Visual Feedback Design

When preparing a meal, visual perception could be used to aid the user by displaying a progress bar to show how cooked a piece of meat or dense vegetable is. This is somewhat possible using devices like a sous-vide or digital meat thermometers (see Figure 2), but they require specialized equipment and generally do not provide much in the way of an estimated time to completion or progress. This can greatly help a cook orient their tasks when they know a food is getting close to being done rather than rely on a timer or food color, which may provide a false sense of "done-ness".

Figure 2 - Left: a barbeque meat thermometer (Source: Amazon).
Right: a sous-vide (Source: AmazingFood).

3.3 Cooking Auditory Feedback Design Challenge

To utilize auditory feedback where it typically is not used in cooking a meal, a device such as a "mini" smoke detector could be implemented above a stove that just warns a user when food starts smoking (generally a bad sign when sautéing or baking). This is the same concept full blown house smoke or carbon-dioxide detector but would be designed in a way to not be so alarming and be less sensitive – maybe making beeps at the same loudness as a microwave timer while having some tolerance for a little smoke. This way a user could intervene before

food gets completely burned or setting off a proper house fire alarm besides just using olfactory senses or visual senses.

3.4 Cooking Haptic Feedback Design

Haptic feedback could be used in a new domain within cooking by making utensils vibrate when things they touch are too hot for human consumption. For instance, after food has been removed from a stove and left to set, serving tongs or ladles could be equipped with thermometers and vibration motors that make the utensil vibrate if the food being served would injure the person consuming it. The same concept could be applied to things like coffee mugs: they could vibrate upon being lifted or tipped if the liquid temperature would scald the user.

3.5 Bonus Sense – Taste Transmission

A farfetched, but very useful perception that would be excellent in cooking is a device that could let a user taste what the content in their cooking utensil tastes like (think of it as like a magical RFID-enabled Willy Wonka gum or lollipop). This would be useful in that it could provide real-time feedback to the user how tasty their food is, even it is just basic capabilities like sweetness, saltiness, or spiciness. The user could "virtually" taste their food while it is cooking without needing to wait for the food to be safe eat such as with raw ingredients, when the food is too hot to consume, or simply in a place inconvenient to taste (such as in a smoker or in an oven). Inadvertently, this probably could be used to satisfy cravings without needing to consume real food in an alternative use case.

4 QUESTION FOUR – DISTRIBUTED COGNITIIVE LOAD FAILURES

4.1 Tip One – Multiple Modality Violation

An interface that I use in my life that consistently violates tips for reducing cognitive loading is simply just tightening my water bottle lid. Despite its simplicity, there is often no other cue that the lid is tight than just feeling resistance. This often leads to me not tightening enough or torquing it on enough in bout of frustration (after soaking something on my desk) to the point where I cannot get it back off easily.

To alleviate this violation of the first tip, multiple modalities could be incorporated into the water bottle lid by introducing some sort of ratcheting lid – like

how some fuel tank caps work on internal combustion engine cars. This lid could simply start ratcheting after reaching the correct torque, providing feedback to the user via vibrations from the lid and noise from the ratchet itself alerting that the lid is secure.

4.2 Tip Two – Complemental Modality Violation

At my company, we usually need to take required trainings every year via standardized webforms on the company website. These can include topics like fire safety procedures (for buildings we do not use anymore) or IT security policy awareness. The problem with these trainings is that their web interface overloads the user with too many modes of information at once by default. Basically, the web interface has a prompt to read, a couple buttons to go back or forward, an occasionally multiple-choice form, and audio that describes the prompt. The real annoying thing about this interface is that the audio starts auto playing when the page loads, and the audio only summarizes information on the page – it does not actually read the prompt or questions word for word. This not only distracts users from the content they are trying to learn but can frustrate users to the point of just skipping ahead instead of re-reading content.

This complimentary modality problem could be solved by instead having the audio describe the prompts word for word, or simply turning off the audio by default. Better yet, the trainings could just be videos with optional captions with occasional quiz prompt pages like how ED or Canvas operate.

5 REFERENCES

1. BBQ Guru Store. (n.d.). *DigiQ BBQ Temperature Control, Digital Meat Thermometer, Big Green Egg Cooker or Ceramic* . Retrieved from Amazon: https://www.amazon.com/BBQ-Temperature-Control-Digital-Thermometer/dp/B00UICE39Y
2. LearnCAEBlog. (2017, August 27). *Beta CAE ANSA Video Tutorials*. Retrieved from CAE Validation Engineering: https://learncaeblog.wordpress.com/2017/08/27/beta-cae-ansa-video-tutorials/
3. Logsdon, J. (n.d.). *Exploring Sous Vide*. Retrieved from Amazing Food Made Easy: http://www.amazingfoodmadeeasy.com/info/exploring-

sous-vide-email-course/more/how-to-prevent-sous-vide-bags-from-floating