Manipulus

//
//
// *Abstract*

My thesis proposes a description of interactivity as an emergent quality of manipulability. I define manipulability in terms that can be the target of specific, quantifiable, design decisions. A series of studies investigates its potential use in creating novel variation and whether it coherently describes both perceptual and computational qualities.

I deal strictly with single-user, screen-based interaction. I also make a number of informed assumptions. I assume that an active encounter with an interactive artifact is perceptually distinct from passive observation of its use — that visual, kinetic, and aural qualities do provide a full, formal, description of the experience. I also assume that if something relatively un-interactive can be made *more* interactive, then my model might also be relevant for more complex interactivity. So while I posit that all interactivity might be described in terms of manipulation, I limit my investigation to the very simple.

This investigation centers on the creation of a series of studies; in each I attempt to apply my language of manipulation both perceptually and computationally in order to create novel interaction from otherwise simple items and identify where the definition is ambiguous, logically problematic, of computationally non-applicable. These studies expose both opportunities for revision, more sophisticated questions about the nature of interactivity, and a growing code library that provides computational equivalents to statements about manipulability. I have also created a very simple GUI allowing for some limited direct manipulation of manipulable qualities.

//
//
// *Introduction*

Need a brief, open, definition of interactivity or interactive artifacts?...

While this paper will focus on specific view of interactivity, I'll begin with the more general and inclusive definition provided by game design Chris Crawford. Crawford defines interactivity as activity between at least two actors — specifically where one is a computer and one is a human — who alternatingly "listen, think and speak." (5) The terms 'listen' and 'speak' are representative of sound, visuals, movements, or any other action observable by the other actor. I'll limit my discussion to those artifacts to those that involve only a single person and a computer, and where the computer 'speaks' through screen based visuals.

This definition includes a vast domain of artifacts from complex games to calculators to scroll bars with the caveat that while all these artifacts may be examples of interactivity, they need not all be viewed as equally interactive. Each artifact, however, can be said to posses some form of 'Dynamic Gestalt' (Löwgren and Stolterman 53)— our perception of them unfolds over time and as a consequence of our actions in relation to the artifact. The complexity of this dynamic gestalt may very greatly.

I'll discuss this later

While I'd like to focus on artifacts that benefit from direct encounter, this isn't to say that some aspects of an interactive artifact may be understood without direct encounter (Lopes 101). Even a passive encounter with representation of a finished artifact may allow us to understand all or some of its qualities. Process methodologies that utilize paper prototyping, scenarios, or story boards have been shown to provide valuable — if general —insights in relatively short order. It's also possible an interactive artifact's overall perception, meaning, or significance is dominated by non-interactive qualities such as its visual, kinetic or narrative qualities which may be quickly formed and experienced using existing methods. Some interactive artifacts require prolonged use-time in order to completely understand the aesthetic of their use (Dourish). In these cases the difficulty lie in the time necessary to experience them opposed to the time needed to create them. Lastly, some interactive artifacts may be understood through contemplation of use opposed to actual encounter. Löwgren explains:

Paper Prototyping
ID book

> "Parafunctional design is generally appreciated in three steps, starting with a simple recognition of the product and its intended function, followed by a brief period of frustration at the obvious inappropriateness of the intended function and only then a sudden insight (the »a-ha« moment) when you realize what the artist-designer wants to make you see."

Where an encounter has the most impact is in cases where interaction with the artifact extensively utilizes direct manipulation. This kind of interactivity is distinguished by a constant change in relation to constant input. Direct manipulation is noteworthy in that it supports both effective and enjoyable interactivity (Shneiderman). Unfortunately, this type of interactivity can be relatively difficult to design.

[citation needed ?...], in comparison to what?

A screen based interactive artifact in the most technical sense is a continuous, interruptible, process running on a computer that controls the creation of visual forms varied in response to a context beyond the

program. Programming languages provide the precise detail needed to annotate these processes in a manner the computer can execute. While often an immutable and essential task in creating the artifact, creating and modifying these programs to produce desired novel[1] results is both difficult and time consuming compared to the traditional sketching methodologies (Buxton 97).

Some methodologies aim to separate and isolate programming from the 'design task at large' in order to minimize the difficulties in programming — specifically to avoid a lack of conceptual integrity and its corresponding ambiguous and costly implementation needs (Brooks 42). This view emphasizes a view of programming as the production work of a pre-formed idea (Reas et al. 25). The aim is to minimize the risk of complicated development issues by strictly defining the needs of the program. This was the very explicit goal of the 'waterfall' development methodology (Sharp ??) and now, more loosely, a complimentary goal of user centered design: Programming is difficult and time consuming, do it as late as possible (Sharp et al. ??). In game design instruction, the programming problem is sometimes avoided by emphasizing the creation of board games over digital games — in these cases a student/designer must still create an algorithmic system but with the leniency that it need only be executed by people opposed to computers.

While pragmatic in concept, this approach is both problematic from a development standpoint (Brooks 264) and is at odds with traditional iterative practices from graphic and industrial design. These methods, as Colin Ware describes, emphasize a designer's repeated encounters with evolving manifestations of their ideas; each iteration modified in order to balance the complex requirements of audience, subject material, and medium. (In the case of game design, analog prototyping simply avoids the medium completely. (Brathwaite) While the visual qualities of an interactive artifact may be easily prototyped using the designers intuitive visual problem solving abilities and drawing skills; rapid prototyping, or sketching, of the interactive qualities of these visual forms is reliant on a designer's programming skills, and perhaps more importantly, their ability to identify and abstract potentially complex behaviors that may be central to the artifact (Gingold, "Catastrophic Prototyping").

[citations: I'm sure I know more places that talk about the need to break problems down...]

Promoting the development of programming skills is another approach. Attempts at helping students overcome conceptual hurtles inherent in programming is at least as old as the LOGO programming language created in 1967. Since then there's been numerous attempts to make programming more accessible (particularly for artists and designers) including software like Hypercard, Director, and MaxMSP, and languages (or programming libraries) like Lingo, Processing, and Open Frameworks. Each of these, while achieving a variety of successes and support from communities, re-encounter similar difficulties. Alan Kay in his foreword to *Watch What I Do*, explains how even simple scripting languages represent a less than ideal learning investment. *"1) Users still*

For practical purposes, this is also the approach I emphasize for undergraduate designers new to interactivity.

Brooks also outlines difficulties

---

[1] Many systems exist for building interactive artifacts from pre-existing components and/or pre-existing behaviors. However, as the screen space is effectively infinitely plastic, these solutions often represent a limited realm of possibility and promote convention over innovation.

[citations needed?!]

need to make use of this term consistent. Should also define that this includes artists etc.

I was introduced to one interesting one very late on: Interaction Gestalt and the Design of Aesthetic Interactions : Youn-kyung Lim, Erik Stolterman, Heekyoung Jung, Justin Donaldson

[Elements of Design and Rowena Reed Costello] check out Graphic design: The New Basics

This section should read:

- Ways of defining interaction (interactivity as a quality of form, interaction as the act of engaging with an interactive form, a subjective experience that is shaped by the interactive 'qualities' of the artifact).
- My definition, and why
- Conflicts with existing definitions
- The 'thinking computer' problem...
  - We attribute intelligence to computers
  - Encouraging _this_ behavior creates problems.
  - We want to understand consequences as the result of our actions. We move towards assimilation
  - We should deign systems to encourage this assimilation behavior. The metaphor is irrelevant so long as we can actively develop an understanding of the causal relationships between our actions and the results... Designing something to be understood without use seems...backwards... in the one use situations we have to design for what they already understand...
  - (Create one u-scan that works the best. Put it with three that are designed for use once...)
- Implications of the 'response to my actions' view.

Need to dig up my aesthetics reading on different types of definitions.

have to learn the arcane syntax and vocabulary conventions of the language, and 2) they have to learn the standard computer science concepts of variables, loops, and conditionals." It may be that the utility of programming is tied to the qualities that make its mastery difficult.

While strict in syntax, programming languages are generally highly expressive. Their abstraction allows writing processes that read input and control output in a variety of contexts unrelated to the visual and interactive problems a designer of screen based interaction is required to deal with. Language like processing mitigate this by hiding instructions unessential to a designer or artist while providing instructions with more immediate visual consequences. This allows designers to affect screen visuals more quickly and restores some of the continuous iterative process. As interactivity requires form, these types of languages will be invaluable for both full projects and in educational contexts.

Similarly, a programming language that provided designers a brief but powerful instruction set for affecting an artifact's *interactive* qualities would be desirable. This would require a definition of interactive qualities abstract enough to account for their perceptual nature yet specific enough to allow for their expression in computational terms. The lack of one is echoed in game design; Game designer Doug Church voiced an open request just such a language. The lack of such a language drove Greg Costikyan to write "I have no words and I must design". Jesse Schnell notes that in the absence of clear definitions
we make do with a variety of lenses (24) — though he points out the the lack of clear vocabulary does not and should no stop the act of creation.

While there are many definitions of interactivity, to my knowledge at the time of writing there is no broadly accepted language with low level detail comparable to that found in traditional design languages concerning gestalt forming components such as point, line, plane, color, texture, volume, and surface, and the relationships between them.
Such a language would be beneficial in variety of ways. Firstly it would help with critical evaluations of both existing finished artifacts and intermediate prototypes. A decisive language would help provide designers with specific decisions they may make in order to effect desired change in an artifact. Ideally it would also help designers identify and prioritize specific aesthetic problems. Lastly, better tools for the design of interaction would benefit from a more clear idea of what activities and choices are most beneficial to facilitate.

Creating such a language is particularly tricky for two reasons: interactivity is predicated on form (Svanæs 5) and the forms of the screen space are — for practical purposes — infinitely plastic.

I believe this lack creates difficulties in instruction, evaluation, and tool design. This thesis will propose how a language of manipulation may serve these purpose in spite of the difficulties the screen presents.

//
//
//: Interactivity

Until this point I've used a definition of interaction that was relatively inclusive, discriminating based on qualities opposed to the degree in which such qualities were present. While such definitions lump together the interactivity of a button and a blockbuster AAA video game, they allow for a range of judgements to be made about the *quality* of the work. i.e. Is such an artifact a *good* example of an interactive artifact. These kinds of definitions also allow for specific formal qualities that a designer may manipulate. For example, when describing interactivity in games Katie and Zimmerman define four categories of interaction that range from activities more accurately described as passive observation (the interactions of two colors in an image) to physical manipulation, to the artificial structures and limitations on choice imposed particularly by game designers, and finally meta-contextual activities that surround game play itself (59).

While Salen and Zimmerman's description identifies the more topical area at hand — *artificial* relationships and restrictions that give actions meaning — allowing for better discourse about the act of *designing* games (or perhaps any interactive artifact) it may be difficult in a screen space where the artifact emerges from many levels of abstract artificial constraints (Gingold, "Miniature Gardens & Magic Crayons") to define where choices lay — outside of the specifics of the development environment. It may be useful to have a language of interaction such that an uninformed person may evaluate the artifact at hand without knowledge of the distinction between an auteur's decisions and the interactive qualities of the materials they began with.[2]

In contrast, evaluative definitions utilize a threshold that some entity most pass to qualify as an example of such. For example; a digital image may, by descriptive standards, consist of only a few pixels, but may not qualify as an 'image' by most uses of the word. This is useful in that the significance of a digital camera may not be apparent until it can achieve a certain level of resolution. Malcom McCullough makes a similar distinction in *Digital Ground* between items which are interactive and those that are merely operable (20). In the context of his book, the distinction is important in order to separate those items (such as elevator buttons) that may cast interactivity as 'nothing new' and the type of era changing interaction that modern networked computers facilitate.

DISTINCTIONS BETWEEN 'TYPES' OF INTERACTIVITY BASED ON MORE OR LESS INTERACTIVE' The distinction between operable an interactive is an interesting place to start.

McCullough's definition may stress cultural significance, but it implies a starting point for my investigation. In making a purposeful distinction between artifacts that are merely 'operable' from those that are 'interactive' there is an implication that the two categories have a perceptual similarity to begin with. That an interactive artifact is ideally more interactive is argued by Dominic Lopes "Philosophy of computer Art" (98). However, he provides no lower limit to the interactivity. I'll posit that as an operable artifact is in some sense a 'less interactive' interactive artifact, an interactive artifact may be a 'more operable' operable one. As such it may be beneficial to study whatever interactive

---

[2] How something like the intentional fallacy might begin to collide with the criticism of screen based interactivity (or, more interestingly, non digital works like board games where the de-facto 'experience' is predicated on some understanding of authorial intent), is beyond the scope of this paper however.

qualities an operable artifact may possess in order to better understand more complex ones.

For my purposes I think an interactive work may be evaluated via the perception of an encounter with it, that some artifacts may be characterized — in some universal manner — as being more or less 'interactive', and that it may be modified in such a way as to make it more or less interactive. It is widely held that basic perceptual and cognitive abilities are biologically contingent (Norman, Raskin, Ware), and as such, a descriptive language of interaction that centered around what we perceive during an encounter — beyond passive visual and kinetic perception — should at least be possible. I will propose one such definition that may be used to describe both simple, "low level" interactivity and interactivity that is, at minimum, more complex, or "more interactive".

A definition of interactivity that contains specific descriptive elements based on common perceptual phenomena would be useful for a number of reasons. Chris Crawford formulates one such definition in order to better critique video games. He states that interaction is made of two actors that alternately speak, think, and listen; and the ideal system balances these activities between the two actors. From this, deficiencies in games (or any interactive artifact) be ascribed to a deficiency — or over abundance — in a particular area of the loop. What was an interactive problem becomes a 'listening' problem for example. In such language, the overall success of the Nintendo Wii and its motion controls could be ascribed in part to their advancement of a system supporting increased 'listening' to balance out the increased 'talking' of photo-realistic games at the time.

This all somehow needs to move to the end...

A similar thought seems to be similar thoughts advocated in Activity Theory, but my knowledge of the domain is entirely second hand...

The idea that the computer alternately acts is the bases of a perceptual assumption. It does not act, it shows us the results of our actions, that given enough complexity, give it the appearence of 'thinking'.

As operation and interactivity may exist on a continuum, I posit that an interactive artifact may be a kind of operable one and an operable artifact, if adjusted in some fashion, may become more interactive. Furthermore I'll define Interactivity specifically as an emergent quality or a collection of simpler operations. The characteristics of these operations may then be modified (by a designer or by circumstances) in some way to become more interactive. The question then becomes; what are the qualities of an operation, and what are the qualities of a relationship between operations.

This directly conflicts with the most general definition of interaction; that interaction is "reciprocal action", and requires one actor — typically a computer — to alternately act on the other party — the user; and that this reciprocal action is the primary quality that makes interactive artifacts unique. My definition reframes the computer's "acting on the user" as an individual encountering the results of their actions.

This framing may be somewhat unintuitive, but potentially useful none the less. In this sense, more complex interactivity may be defined as a more complex relationship between a user's actions and their consequences. By complex I mean that the relationship is more nuanced, subtle, surprising, demanding in attention, and most importantly, continually intelligible — i.e. further action continually results in a better or more sophisticated understanding of the artifact. By

complex I do *not* mean the relationship is less intelligible, or more chaotic, or more arbitrary. The idea of continued intelligibility rests on the premise that most people are naturally, to the best of their ability, driven to understand the changes of a dynamic visual system in terms of consequences-to-their-actions. Svanæs shows that (with at least simple systems) individuals initially describe unexpected events as contingent on another actor, and through use the seat of 'actor' gradually becomes situated within them (157). Assuming a user's natural inclination is to understand an artifact's behavior in terms of their own actions, and that this activity is inherently compelling, an ideal interactive artifact might, baring anything else, strive to strike a balance between seemingly independent behavior and behavior completely directed by user input.

Ug, this is a good place to cite the learning argument. "This is akin to learning, and has been proposed before." However, while our understanding of the relationship between action and consequence may be most observable at action-reaction an abstract level, the semantic qualities of the artifact (narrative, forms) are undeniable important. Tetris and the slave block stacking game may have the same mechanical relationships but are hardly the same. (TLDR; game essentialism is spltph)

I do not mean to say that people who encounter interactive artifacts consciously understand their interactions in this sense. It's commonly observed — particularly in the case of visually sophisticated, semantically rich, or more interactive artifacts — that people attribute all kinds of personality qualities to the artifacts. It's even noted that 'better looking' artifacts are perceived to 'work better' (Norman, "Emotional Design" 17).

Furthermore, while a user may view the artifact differently, this does not negate any value the idea may have for a designer. Similarly, in the visual arts a viewer need not have an understanding of color theory in order to view a work, but its utility to the designer is hardly diminished. In contrast, the view by a designer that an interactive artifact acts (let alone thinks) may actually have have negative consequences on their design decisions.

Don't hide the model, describe it. Building the interface first begs us to remediate what we know and interferes with the creation of tools that utilize actual computational abilities.

Disparity between a user's perception of an artifact's 'intelligence' alongside the artifact's actual capability to respond or act appropriately is a noted failing in a number of interactive projects. (Sharp et al.) This phenomena also has parallels in video game design where a dissonance may exist between the sophistication of a character's visual presentation and that of their behavior. The phenomena may be considered an impetus for "Max payne cheats only" by artistic duo JODI. In the work the digital character assets of the game "Max Payne" are digitally modified to produce a sometimes grotesque if not other-world views that echo the dissonant intersection of the human world with the foreign space of computation.

ntentionally destroying what Bolter and Gromla describe as the transparency

On a more abstract level this is the same problem with the design of interface metaphors. Oftentimes created to facilitate a user's intuitive grasp of an artifact's working, they can be counter-productive when the structure fails to match the perceived façade. Worse yet is when perceptually accessible part of the artifact is seemingly unrelated to its function. This disparity between the internal workings of systems and their interface led to the advocation of designing from the interface backwards, or more appropriately, from the user backwards. (Cooper ??, Norman, "The Invisible Computer" 23). This position rightly emphasizes the design of interactions in response to the user as whole person and to avoid forcing them to bend to the design. Without a clear relationship between design *goals* and executable *decisions* I wonder if this view unintentionally emphasizes solutions built on what users already

understand — or worse, on what designers understand — opposed to potentially novel solutions that may work better or are at least more enjoyable. It may be that the design problem may not be best approached as "designing the interface first" but by designing the interface to communicate or express the system's function in a manner that rewards continued use by providing continually compelling interaction.

If a designer can't design compelling interaction without the benefits of a utility to drive how will they be able to make a 'good' interface?

Ideally, a designer should be able have intuitive understanding of their medium in order to solve complex problems. While this generally comes from experience; it'd be ideal to focus on problems that meet their developmental level and allow them to practice solving problems that continually reoccur. Also, as significant utility will encourage use almost regardless of the quality of the interface, the challenge for a beginning student designer should be to create engaging interaction sans-utility; and they should do it through careful investigation of the variables at their disposal. The definition of these variables and their preliminary investigation begins in the next section.

(((General definition of manipulation, and its general ramifications.)))

Activity theory uses operation as the activity below conscious perception. So in this sense, manipulation is actually **more** complex, or closer to interaction.

What distincquishes manipulation from action? hmm..

//
//
//: Manipulation

In describing a continuum between interaction and operation, I'll use the term 'manipulation' to describe those actions that are both on the continuum but are the *least* interactive[3]. The definition of manipulation is to handle or control, typically in a skillful manner. The root of the word being from the latin *manipulus* or 'handful' and *manus* hand. (It is also the root of the roman *Maniple,* a division of the army; one considered to be a 'handful'). The term manipulation is also used metaphorically to refer to the control of particularly complex items, and even in a social context — the most 'interactive' of settings. Regardless of the specific use of the term 'manipulation' there's reason to believe that our understanding of the phenomena is built on the less complex, body-centric, understanding of the term (Lakoff and Johnson). Lastly, as manipulations are predicated on causal relationships between items, these relationships themselves should have qualities that are in turn manipulable.

My definition of interactivity then requires a strict definition of manipulation. For my purposes I'll define manipulation as: The intentional bringing-into-alignment of some perceived quality of an entity to that of an intentional value. In other words, manipulation is an action with several criteria:

• There must be an actor with intent.

---

[3] The term 'operation' is used in Activity Theory to describe those actions that are accomplished without conscious thought, and make up more complex guided actions.

• There must be an observable, external entity.
• The entity must have some observable *quality* that may change.
• The observer's locus of attention is on this change.

For example; turning my coffee cup so the handle faces me, the entity is the cup, the property it's rotation, the intentional value is a preconceived rotation where its handle faces me, the manipulation as a whole the act of transforming its rotation to that of my ideal.

While I believe this definition useful, it is problematic in a number of ways that I will spend the remainder of this section addressing. Firstly there is no available method of quantifying 'intent'. It will then be best to discuss the point at which an actor's intent manifests (i.e. input). Next, and perhaps most problematic, is the requirement for a persistent entity with observable qualities. Not only are people highly capable of perceptual shifts that reframe fundamental perceptual starting points as figure and ground[4], but the screen space may present visual forms with computational structures divorced from our common perceptual understanding of them; In other words, the entity-ness of screen based forms is highly dubious. Lastly, detailed understanding of the mechanisms or manner in which a user's locus of attention changes in response to action is presently beyond the scope of my research.

Not convinced...

In the coffee cup example, the manipulation is composed of changes in any number of perceivable entities; an arm, fingers, the cup's saucer; without being the locus of attention these do not count as manipulations however. That our locus of attention may move between elements or actions of a larger task a given. If our attention should move from one point in this causal chain to another it may be ambiguous whether this should be described as two sequential manipulations, two overlapping ones, or one manipulation with some allowance for the specifics of our attention. The distinction I'll leave open for now.

Understanding how we process such hierachical actions would be really useful to know. Further research in activity theory would also be beneficial

One thing that might be worth distinguishing is a area of a screen artifact that may draw our attention through visual or kinetic means, and the parts of the visual that we associate with our ability to act. Game designer Greg Costikyan makes a distinction between the elements within a game system we control directly (Tokens) and those things we manipulate through the use of the tokens (Resources). This is valuable as a mental tool for design or evaluation, but more fascinating is that, in practice, what qualifies as one or the other may change, whether literally or from a subjective point of view.

There's tons of examples of this in games and interfaces. I could talk on end.

In the sketch *Twins 01* two cursors respond directly to input. However, one cursor vibrates when the user gives no input. Once a user provides input, the behaviors of the cursors switch, the one that vibrates is calm, moving like a normal cursors would, while the other, previously still cursor, follows but with an overlay of wiggling movement. While the vibrating cursor may draw our attention, the feeling of 'under our control-ness' seems to belong to that entity exhibiting behavior most similar to

---

[4] See Wittgenstein's duck-rabbit image or other classic optical illusions such as the old lady / young lady image.

our own; in this case the still, or continuously moving cursor. This change where the perceived agency moves from one element to another I describe as a “Token Shift”.

One common and peculiar type of shift is inward, where an action or manipulation is interrupted by an otherwise un-expected turn of events — or a misbehaving relationship in the chain of causality — forcing our attention to it. An older study of mine, *token switching* demonstrates this. In it a grid of cursors are variously ‘activated’ and ‘deactivated’ dependent on the (invisible) system cursor’s location. An active cursor is tightly mapped to input, and a deactivated one moves to its original place in the grid. The objective result is that as the user moves a mouse different cursors will respond in the manner expected; the subjective and experiential result is generally one of unsettlement. (At the time I found it noteworthy that the experience of controlling the system is distinct or at least more acute than the experience of passively observing its use.)

An aspect of this phenomena can also be partly described in terms of a shift in a person’s sense of agency. Dav Svanæs describes how the sense of agency is subjective and subject to change even when the system’s behavior is constant. In his studies he observed a shift in users’ perception of a simple interactive system — at least in the language used to describe the behavior of the system — their descriptions of the behavior changing from more independent (the computer acts) to dependent (I act) the more time spent interacting with the system. In other words, through use the user’s “loci of attention” moves through a continuum beginning with their body and ending with the changed element in the screen. (157)

Picture?

The study *TokenChain* is an example that allows users to instigate shifting. In this case though the shifting is ‘outward’. The ability to manipulate the cursor at the center of attention is never removed, but instead the consequences of this manipulation are extended; the cursor’s status as the focus of the common ‘point and click’ gesture is shifted through one element to another. As the chain extends, the focus of our attention moves with it. While the entity of the manipulation changes, the specifics of the relationships (input to output) do not. In contrast to the previous study, the experience of use is not unsettling, and in fact is barely noteworthy.

The two types of shifts I’ve described, ‘inward’ and ‘outward’ seem dependent on a perceived (if not actual) causal relationship where one element affects those further down the causal chain. In *token switching* attention is pulled towards an element that would otherwise be acted-with unconsciously. In this case it might be said that the change in the actual causal structure outpaces or disrupts our expectations, whereas in the *TokenChain* the reverse may be true; the change in the actual causal structure follow behind the change in our locus of attention and reinforces expectations.

That this ‘shifting’ is in some way an experiential phenomena with aesthetic dimensions — it is capable of prompting some kind of feeling — is clear; Its specifics less so. I am unsure, for example, as to whether to described it as a single manipulation with a changing entity, or two

separate manipulations that happen in sequence. An answer would be best based on a better understanding of both the manner in which our locus of attention changes, whether moving 'laterally' between unrelated elements in the same context, 'hierarchically' between causally related elements, or if such distinctions are even tenable. The specifics of this are of great interest to me but beyond the scope of this paper. It is enough here to note that at one point what was once a manipulation of one thing at the locus of our attention is now a manipulation of another, and that despite being contingent on subjective perception, these perceptual shifts can be instigated by the system itself.

Manipulation and entityness.
May an entity transform?

Tension and release in behavior.

There are several known visual cues that we use to distinguish one thing from another including proximity, connectedness, and sympathetic movement. Things that are close together often go together — or at least affect each other. Things that move together are often seen as part of a surface — typically on a solid object. Kinetic cues and otherwise 'static' visual cues may easily disagree; for example, in an instance where two distinctly separate dots move in unison — imagine a dark night watching a car's headlights from afar. While this conflict in cues might be cause for a little tension, it would hardly be described as off putting or uncomfortable. However, when the simultaneous movement of two, visually distinct, dots are under the control of a user, it's possible — in situations where the distinctness of the dots would imply a difference in behavior — for any subtle tension to apparently increase. In *twins_02ab*, either a bar or two 'connected' dots may be dragged within the confines of the space. When the bar's movement is limited by the constraints it seems perceptually neutral, unremarkable, perhaps natural. Whereas when the two dots are stopped by their constraints there seems to be at least a momentary tension, as if we expect the two elements to behave independently. A following study, *twins_02a* similarly contains two circles that will move together when clicked and dragged, however each is confined to half of the space. Once one dot collides with its bounds the dot's movement will become restricted whereas the other may continue to move freely. Once one or the other exhibits some independent qualities, there seems to be a release in the visual tension. It would seem that while our mind may give the kinetic and manipulable qualities of a form or forms some precedence in regards to its status as an 'entity', it does so perhaps grudgingly or with reservations.

Another study is built with similar behavior. However the un-clicked circle has noise introduced into the mapping between the mouse and its position resulting in a wiggling when dragging. At various times, its direction of movement will be either complementary, opposed, or tangental to the direction of input movement and also to varying degrees. The result appears as an ambiguous causal relationship where the tightly mapped dot affects the wiggly one. However, at times it appears as if the dragged dot should be responding to the wiggly one, creating slight tension when the dragged dot doesn't conform to expectations established by the visual relationship. This is somewhat more prominent in *twins_03* where a line is shown connecting the two while dragging.

In *twins_04* a collection of dots is used. When a dot is clicked, other dots will move somewhat sympathetically but with limitations. The initially clicked dot is tightly mapped to input and responds immediately and

exactly. Another selection of dots will move as if tightly mapped, except they will move in fixed increments of distance (that of their width), and only if the velocity of the input movement is above a certain thresh hold. They have the appearance, or feel, of being ‘sticky’. Another selection of dots moves more continuously, but lags behind mouse input movements. While the clicked dot always responds immediately and continuously, the focus is drawn to the collection of ‘sticky’ dots and their somewhat haphazard response. I might describe the overall effect as a ‘tease’.

It may be reasonable to say that in a more complex artifact with many things responding to input, we look for or notice what is at the tip of our conscious control, and that the target of a manipulation may be influenced or redirected by the manner in which the system responds. In the cases like *TokenChain* where a user may consciously switch their objective point of manipulation from one element to another it may be said that a form’s manipulable qualities may themselves be the target of a manipulation.

Or you could say there is a gulf between action and results.

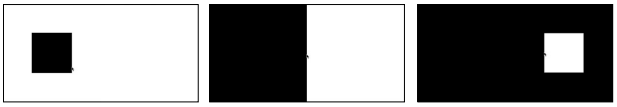
Lastly, in the screen space there is nothing a priori manipulable. As screen visuals and their behavior are materially the result of complex mathematical algorithms, some have argued that this constitutes the essential reality of the artifact and that anything beyond that is ‘window dressing’ (Koster). In reality, the specifics of the *inner* workings of these creations may be unobservable but they may have no baring on how they subjectively understand them or attempt to interact with them; the math and logic used to display a black square on the screen are secondary to our experience of it as such, unless its computational description have direct baring on what qualities of it are manipulable.

For there to be manipulable form, the form — and the manner in which it can transform — must be created. This is typically done through the combination of mathematical expressions and programming logic that defines, in the end, a visual form commonly referred to as a parametric form (McCullough 168, Reas et al 93). Such abstract forms have quantifiable parameters that define a range of potential observable forms they may take on. The continuum of forms may be referred to as a possibility space.

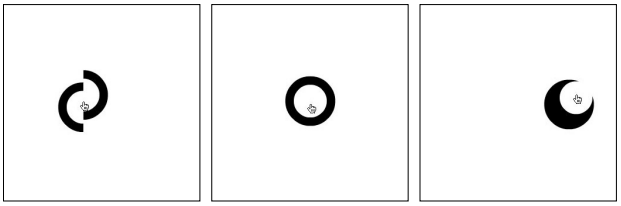
It is entirely possible for two parametric forms to share one or more specific overlapping expressions. At these points a visual form’s manipulable qualities may be ambiguous. It may also be possible to switch between two different parametric forms. The quality that was once manipulable disappearing in place of another quality (or even an entirely different entity) without any visual discontinuity.

In the study *interactive figure ground* a rectangle is divided into half white and half black. It may also be described as a white square on a black background and vice versa. By moving the mouse the system changes from one structure to another, switching between them at the point of overlap (where the rectangle is exactly half black and half white). The study *Fidelity* is another, slightly more complex, example.

interactive figure ground



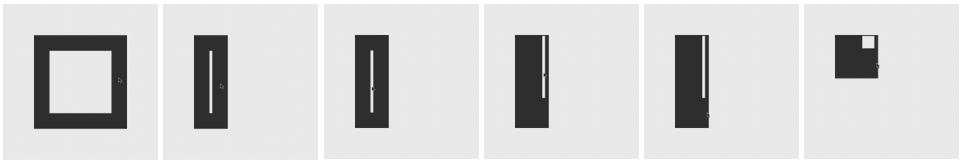
Fidelity



It's also possible for the possibility space of two or more parametric descriptions to overlap at more than one point; even overlap completely. *Ring Box* allows a user to manipulate a form in one of two possibility spaces; one where the individual 'bars' of the form can be moved via a click and drag, and one where the negative space in the middle may be moved as if it were a solid form. This phenomena is found in almost any piece of software from word processors — allowing for the manipulation of their content as-language (in the case of typing) or as-image in the case of setting type face, type weight, margins and other formatting variables — to 3D modeling programs that support the manipulation of form in terms of points, lines, or surface.

Bigger!

ring box



In these examples there is a clear continuity of controllable transformation, but if there's a specific entity being manipulated it is somewhat ambiguous. The most general description would be that these are simply parametric forms. However, these situations where the same or similar action has different consequences could also be referred to as modes (Raskin 37). Furthermore they might also be described as two forms that support instantaneous "transcoding" (Manovich, "Language of New Media", 45) in that the perceived form is bounced between two different mathematical or numerical models (though its possible for a form's mathematical underpinning to remain constant even while a person experiences a perceptual shift). In my definition each study might be described as supporting a 'token switch' between two manipulable

Ian Bogost also has some interesting writing in **Unit Operations** that uses a language of "fidelity" to describe how actions in a space reaffirm decisions about what constitutes the space in the first place.

forms that have momentary visual similarity, or a switch between tokens coinciding with an overall change in the elements of the artifact.

While we may have a single locus of a attention, it may be possible — if even to a limited degree — to manipulate multiple things at once so long as we can make some attempt to abstract them into a 'unit' (this question I'll return to later). In such a case it may be possible that, given an increase in quantity of complexity, the manipulation of a collection of things might become the manipulation of a gestalt level quality of a larger 'whole'.

A detailed analysis of digital form — being that form predicates interactivity — would seem to be a necessity for a proper description of interactivity; unfortunately the variety of parametric form is limited only by the creator's faculty with math, logic, programming and available computational power. While it may be logical, or colloquial, to in turn give something like "the pixel" material status, the pragmatic view of computational aesthetics emphasizes the algorithms and programming fundamentals — assignment, conditionals, loops, and functions — that change these pixels (Reas et al. 13). Because of this it would be ideal for a definition of screen based interactivity to bridge the language of computation and a perceptual based language of interaction.

Understanding Interactivity

That the manipulable qualities of form share some perceptual similarities regardless of the specific form is implicit in the phenomena being named at all. It should then be possible to describe perceptual or formal changes between forms stemming from a users's conscious action with a common language regardless of the specifics of the form. That we are mentally capable of understanding novel forms with novel manipulable qualities (learning software or new game mechanics) implies an ability for abstracted reasoning about causal relationships. As such, it should be possible to outline various general qualities particular to manipulations that are applicable in the design of manipulable form regardless of the forms' specifics.

Is apparent with every video game that requires users to any number of novel causal relationships. (...and every young designer who wants nothing more than to learn software...)

These qualities should in turn be 'manipulable' in the generic sense. A person who creates or modifies an interactive artifact should not only be capable of describing its manipulable qualities, but should also be able to make and execute decisions about how the specifics of such qualities may change in order to produce a better artifact — if only a more interactive one. In this sense the designer is engaged in a form of meta-manipulation.

How might manipulation be manipulated? Objectives of the Project

If this topic is going to be separate from the studies, I should probably discuss how existing manipulable qualities should be able to be manipulated by a user.

//
//
//: Meta-Manipulation

To speak about designing manipulations in any pragmatic sense at least two things need addressed; A decision on the form being manipulated (and/or the forms available for manipulation) and some quantifiable qualities of a manipulation that may in turn be modified by a designer. The project portion of this thesis is concerned with testing these assumptions:

- *Any defined manipulable qualities will largely compatible with computational expression. I.e. they will be sufficiently specific and quantifiable to express in programming logic, but abstract enough to be applicable regardless of form, so long as the form has some quantifiable quality.*
- *That variations in any qualities will have readily observable barring on the artifact's dynamic gestalt.*

using this wording, the thesis is already proved... I'm not attempting to validate that designing a manipulable artifact is possible, but that my definition might be applicable to any form. As such it should be entirely applicable to one subset of forms.

The most important aspect of a manipulation as defined so far is the requirement for an entity — specifically a quality of an entity — to be acted upon, and the subjective contingencies of its entity-ness. Also, the ability for our subjective understanding of the target entity to change should be accommodated. I'll assume that an artifact that facilitates, encourages, or allows for this behavior will be more compelling than one that does not. As such I utilize a collection of many elements that all respond in order to allow for the perceptual 'target' of a manipulation to ebb and flow in response to the emergent aesthetic. instead of a strict investigation of the mechanic or aesthetic of these shifts I'm instead interested in how minimal variations in other qualities encourage these perceptual shifts or not.

this is similar to the ideal of 'meaningful choice' in game design

Concerning quantifiable qualities, I initially focused on what I felt were two intrinsic and readily apparent aspects; that manipulations are generally not instantaneous, and that there typically exists a quantifiable relationship between the observed change and the change at the point of input.

By my definition a manipulation has a distinct beginning (the formulation of the ideal state) and end (the resolution of the transformation) — the time between lends the manipulation a temporal dimension. The time between the formulation of the ideal state and the end of the manipulation I describe as latency. Latency is typically defined as the time in a system between input and any response at all. This is commonly studied aspect of human computer interaction as variation in latency is easily observable when present, and when above a certain threshold it will diminish, if not destroy, the perception of interaction (Swink 45). By the strict definition there is little room for a decision to be made about latency; the lowest latency possible is generally preferable. In my definition however, it may be used to describe manipulations that take more or less time to resolve. I'll refine my description of a manipulations temporal qualities later, but for now I'll use latency to refer to the time between first input and the resolution of the manipulation.

The second quality stems from the temporal component. As a manipulation takes time the manipulated quality should then change over time between it's beginning and end state. Furthermore, this change may not have a one to one correlation to the input. It need not even be a linear relationship. For example; in dragging a scroll bar, the change in position may easily speed up, slow down, or even overshoot and return. In two dimensions it is more apparent as the movement of an object from one point to another may travel along a curved path — a common trait of natural motion (Johnston and Thomas). Mapping generally refers of the relationship between input and output, but occasionally I find it easier to

refer specifically to the range of values a quality might pass through between its initial and ending states; this range I call the *value space* of the manipulation.

In applying these concepts to a series of interactive studies I was forced to articulate them in a manner that was both computationally meaningful. Alongside the studies I began developing a library that would support the articulation of these relationships with increasing brevity and flexibility. This helped me discover any ambiguities inherit in my definition and unexpected consequences of its application. This library would also serve as the basis for a prototype application. Attempting to describe the language in visual user interface terms helped me understand ambiguities from a perceptual standpoint and identify where it didn't support what would otherwise seem to be a natural choice (such as many-to-many causal relationships).

As the studies have no utilitarian use, nor objective, their use becomes playful, perhaps directionless. This seems at first problematic because intent is one of my requirements for manipulation. However, through use there is the opportunity (and tendency) to form expectations about the responses to input, which seems to qualify as some kind of intent. It may be asked then if a small change in input constitutes an intentional change and thus a manipulation or a single *incidental* action of a larger 'gesture' that itself qualifies as a manipulation. For better or worse I believe the answer in these situations is subjective and, baring context, there is no way for a computer to perfectly discriminate between understanding means and ends. However, the responses to otherwise incidental input might encourage more specific gestures by providing unintended visual or kinetic consequences that a user finds pleasing. Even in designed systems, users can often subvert the expected use or play patterns for the user's own aesthetic ends — video gamers often 'discover' or create new game or play patterns within a system such as "trick jumping". Far from being a deficient, programs that promote unintended emergent activity are often a specific goal of game designers in particular (Salen and Zimmerman 158). Contrary to the seeming specificity, beginning with low level interactivity — with manipulation — may be the best way to create a *general* feel for an artifact's interactive qualities — so long as the manipulations in question are a dominant aspect of the overall use.

trying to say this issue is really a non-issue

THIS

need a better transition...

//
//
***//:* The Studies**

My initial explorations looked at offsetting the entire duration of a 'manipulation'. For example, with zero offset, a cursor would would move in tandem with mouse input, with a larger offset, the cursor would complete the same movement(s) in the same amount of time and in the same manner, but would begin at a later point in time. Furthermore I would create numerous cursors, each with increasing offset. (Latency studies 01, 02.)

In an attempt to touch on mapping I then introduced noise into the relationships between input and position. In doing this I accidentally introduced variation into what I came to call the manipulation's *tolerance*. As the ending of a manipulation would be qualified subjectively, it follows that some results may be 'close enough'; that in turning my coffee cup to face me, it need not be rotated *exactly* ninety point one degrees, so long as its new orientation is functional in the context of the initial manipulation. In sketch latency_03, there is progressively more noise introduced into the both the value space the position values move through, but also the values that they end at. In latency_03a, the latency from the first studies is removed leaving only the noise in the value space and tolerance.

In latency_04 elements with less latency have more noise in their mapping, and elements with more latency have less noise. The results are peculiar in that the elements are both ill responsive (perhaps frustratingly so if it was in the context of some utilitarian application) but are comforting (for lack of a better word) in distinct ways. Tension is created between immediate but ambiguous response vs. clear, but delayed, understanding. I'm not sure if either behavior could be said to better 'echo' the intent or input of the user.

should define or discuss this term earlier? Perhaps discuss now.

Or the desired kinetic results...

05 swaps the mapping between x and y of half of the cursors which otherwise have increasing latency. The cursor with the most direct mapping is easily — and comfortingly — found, but the cursor with the next lowest latency draws attention to itself despite the inverted positional mapping. I imagine we rely most on traditionally defined latency to discriminate between the effects of their actions and otherwise independent events. In other words, causal proximity might be more or most important for establishing our locus of attention than visual or kinetic similarity. This could be an area of future investigation.

This seems to be the consensus elsewhere.

It'd be interesting to forcefully change the seat of identity over the course of a game. This happens already.

Until now, elements had consistently increasing latency. In 06 the difference in latency between the closest mapped element is relatively high, but the difference in latency between it and the next decreases steadily. The initial effect is mostly kinetic, a flurry of action following initial exploratory gestures.

07 is the same as 06, except a property, rotation, is mapped to the 'direction of movement'. Instead of mapping one value directly to another, a property (rotation) is mapped to something more formulaic: the angle between its current position and another position. The distinction between a 'measurement' or more 'absolute value' such as position and a more formulaic or higher order value one like 'angle-to' is, in reality, a arbitrary (as a screen based form, it may be described as entirely formulaic). The distinction is contingent only on the data structures in the programming environment, which may not manifest in clearly observable ways. So there's no real reason to create such a distinction. Here though the distinction between the existent computational qualities and the mental faculty of the artist/designer comes into play. It may be ideal to make manipulable a quality of an element that is clearly perceptual, but has no computational equivalent. For example, the distance between two corners of a box may be relevant to a designer's idea, but have no analogy in the code base at hand.

Parametric objects.

Object as result of an algorythm, or as a defined set of qualities.

This gets into really messy land.

Inversely, the computational environment may support properties that are otherwise unapparent to a person, either because of the level of their programming skill, or simply their creative and subjective way of perceiving the screen space. This tension will be discussed later.

Secondly, the manner in which delay has been introduced programmatically means that any property has not only a current value, but a value at any given previous time. In a sense the manipulations of the delayed cursors aren't delayed so much as they are being driven by a property-as-it-was, ie. the mouse-position-*n*-steps-ago.

In the end, the cursors' rotation gives them an entirely determined, but perceptually independent quality.

At this point I re-factored the code for the initial sketches to make future sketching/coding easier (latency_08). After these changes the system also supported a number of new qualities. This was the beginning of the Manipulus library. A collection of code that attempts to provide computational analogies to my interactive-aesthetic terms of manipulation. I'll describe it in more detail later.

Latency_09 makes use of varied frame rate — essentially affecting latency in the traditional sense. Holding down the mouse button decreases the frame rate, releasing increases the frame rate back to normal. While the tightly mapped cursor was originally easy to spot, as the frame rate for the collection drops uniformly, the cursor becomes increasingly hard to identify with.

In 09a, holding down the mouse button decreases frame rate — or increases traditional latency — non-uniformly. Cursors with more delay have higher frame rates making them move smoothly, while those with less delay have lower frame rates; their movement stuttering. Once the mouse is pressed and held, to me it appears that the smoother moving cursors draw attention to themselves despite being clearly separate from input. *latency 09b* and *latency 09c* are variations on the way stuttering is staggered.

What if stuttering was related to distance?

Quality of mapping (continuous vs stuttered) vs. accuracy of mapping (results of a movement).

Show thinking that got to this.... namely that I wanted overlapping latencies... though it also looks at the tension between visual and interactive cues.

latency study 10 provides for delayed manipulation of one set of an elements qualities (position), but more immediate manipulation of another — rotation. When the mouse button is pressed, the rotation of each cursor simultaneously animates 180°. Triggering the rotation gives the artifact as a whole a certain unity — while otherwise appearing more as a 'collection of elements'. As a corollary to the adage of visual perception "things that are close together, go together", it might be said that "things that act together, go together."

This idea I returned to later in the short series named 'finger studies'. In these studies — built from the initial latency studies — an "invisible button" sits at the center of the screen. In the first study cursors that touch the button change to the familiar "finger" cursor. In the third however, all the cursors change when the real cursor touches the button. Here the cursors are disrupted by the increased delay, but re-asserted in

This tension between visual cues and kinetic/ interactive cues is 'interesting'

what about the second one?

some fashion once a different parameter visual becomes immediately manipulable.

latency study 11 is similar in concept; the rotation and position of each cursor is directly modified by mouse input with the modification of their positions being delayed. The rotation however is driven by the angle to the most tightly mapped cursor position. Here the delay on the cursors' position results in variation in their rotation. In contrast to the previous study, the immediate control over their rotation makes the cursors appear — at least during initial tinkering — to be *more* independent.[5]

Deliberate and VARIABLE response. The cursors seem contextually aware.

There's probably a perception that 'they rotate' more than 'I rotate them'

This may change with use of course if the user utilizes this behavior to achieve specific kinetic aesthetic results.

The twelfth study was an attempt to have cursors driven by mouse movement in both a delayed fashion and by immediate movements. In other words, the position of each cursor is driven by both the cursor's current position relative to the real cursor and the previous position of the real cursor. The results were ambiguous and had unintentional glitches. Many of the cursors with higher delays rapidly flicker between two positions. Having a single value driven by multiple inputs is a technical issue I'd wrestle with later.

When making preliminary prototypes of the 'app'...

latency 13 puts allows the each mappings' latency to be modified by mouse press. Pressing and holding continually decreases the latency resulting in cursors 'accelerating' towards the mouse until they are all moving together. Releasing the mouse button returns the delays to their initial, staggered, values. Lower levels of delay result in mouse-trails.

The objective at this point has been to isolate and explore interactive qualities to the exclusion of visual variables as much as possible. (Avoiding variation in the studies' kinesthetic qualities would be much more difficult, if not impossible). The next few studies (14,15,15a,15b, and 15c) look at how any of the relationships in these interactive systems might be retained while modifying or replacing the visual forms related to them. It may be interesting or helpful — as exercises for a screen designer — to switch back and forth between isolated 'visual' properties and the system's 'manipulable' properties in order to promote both flexibility in thinking and better understanding of the relationships between an artifact's computational and perceptual qualities.

Tie this back into the ontology shifting talked about earlier.

//

Important that the description supports logically consistent behavior.

The qualities of a manipulable form should be able to vary at the creator's discretion, in other words they should be 'manipulable' by both the general and my specific definition. It should also be possible for manipulable qualities to change while the artifact is being used.

The most trivial example would be a box that may be clicked and dragged. The form's position is intentionally changed by user input; while other input (press and hold) intentionally changes whether the mapping of that manipulation is nullified. In this example two different inputs are used to manipulate two different qualities. Just as the same input might

---

[5] It should be noted, that unlike other manipulations, the transformation of the cursor's rotation does not require constant input; a distinction I'll return to later.

be drive the change of multiple cursors, it should be able to drive both a manipulation and the manipulation of that manipulation, a meta-manipulation of sorts. As such it should be possible for the dragging a box to effect how it may be dragged. A kind of interface widget with behavior dependent on its own settings. A simple example is *drag box 01*, where mouse input determines both the position of the box and also the actualization of this relationship.

This could be described visually/interactively much much better

The result is aesthetically banal, but it's important that any definition should support a variety of behaviors. This behavior alone may be described, computationally, in many ways. This description of the interactive behavior may not occur to a user, but may yet be understood once explained.

This relationships might not be intuitive. But with practice it may be possible to evaluate interactions that normally support a plurality of descriptions with the same language.

No one would think this way. But user doesn't have to, the designer has to.

It is problematic: Is the change in the mapping an intentional change? In an effort to consolidate the manipulation of multiple —often abstract— qualities we look at it's constraints as a quality of it? In that sense it may be better to redefine the entities of a manipulation as some distinct element that has observable and variable parameters **with constraints**.

Or a manipulation may be suspended. The iterm results persisting.

What happens to a manipulation deferred? It is suspended, its results persisting.

However, this language doesn't support the next step involving the conditional logic present in any programming language: I**f this than that.**

In the first example the mapping is dependent on the input. In modifying a manipulation it should be possible to also change the property being manipulated. In the simple study *box 02* input changes the boxes's target manipulable quality from position or rotation.

As it should be possible to manipulate the characteristics of manipulation's mapping I created another study where the default 1:1 mapping is changed to 2:1 (the output is half the input) when the box's position reaches a specific threshold. The result is a feeling of the box encounters resistance when dragged the wrong way past a point, like rubbing soap against the grain of a shark's skin. It also had the feeling of pulling something through a membrane. I was compelled to adjust the visuals to abstractly represent de-boning a chunk of meat. One de-boned, the left over form can be dragged with impunity.

As the resulting interaction is at least initially surprising, it's difficult to say that the qualities of an ongoing manipulation are intentionally being "manipulated". However, it also seems entirely intuitive to state — once the relationship is apparent — that the box's "draggability" is a function of its dragging, and that a user who desires to create a 2:1 mapping may do so very intentionally simply by dragging the box to the marked threshold.

Conversely, I'd suspect that a talk-aloud experiment would show that users describe the behavior in terms of the computer, or an element in the system, affecting the interaction, opposed to the user stating that they were "slowing down" the box. My question would then be if it is possible, and under what circumstances, for a users' "body space", to use Svanæs term, would expand to include the notion of "manipulating" the qualities of an ongoing manipulation. Such a conception might be possible but require substantial use time constituting the markings of "expert level" knowledge. Furthermore there is research that suggests that people view action and reaction in a relatively discreet number of combinations of an actor performing an action (Pinker 219). It may be that the appearance of resistance is always considered as the result of "two parties" even if one is inanimate; in such a situation I'd be very curious about how the idea of a locus of attention resolves.

Another distinction that arose in my mind from this study (and from the rotation of the cursors in *latency study 11*) is between a manipulation that requires constant input to bring it to completion —like dragging — and one that may be have a duration but requires only a single, relatively instantaneous, action — like clicking. It seems that two manipulations may require variations in potential input while the the range of values that the manipulated quality will pass through could be identical. As such it'd be worth distinguishing between the value space of a manipulation (the range of values the quality of an entity may go through) and the effort space (the range of input or activity required to achieve the result.

Intent is not anything.
A person must act.

//: The Manipulus Library

It's really a fancy form of assignment. Really, most of this seems like putting computational forms in more human understandable terms.

//

assignment
loop
conditional
function

During the previous studies I began working on an Actionscript 3 code library to support my explorations — attempting to formalize my evolving definition of manipulation.

The first and essential goal of the library was to support a succinct statement linking input to a quantifiable quality in the system. The result is similar to the concept of a pointer, where a variable in a program points to another instead of holding its own unique value. The simplest such a statement might be:

mouse position **->** box position

This statement should be qualifiable by some kind of latency and mapping.

mouse position **->** box position *(with latency "of" [or of a type], and a mapping "of" [or of a type])*

Furthermore the relationship and its qualities should itself be manipulable to support statements such as:

mouse position **->** ` box position.
mouse position **->**" (**->** ` latency)

Such a system should also support chains of relationships (in order to support the phenomena of 'token switching')

mouse position **->** box ¹ position
box ¹ position **->** box ² position

There also needs to be a distinction between an absolute mapping — where the value of a driven quality is a function the current value of the input — and a relative mapping, where the change in a driven quality is a function of a change in the input value.

Δ mouse position **->** Δ box position

Unthinkingly, my experiments almost always relied on such relative mapping. It seems to be more intuitive when things 'move as I move' opposed to when things 'stick to me'.

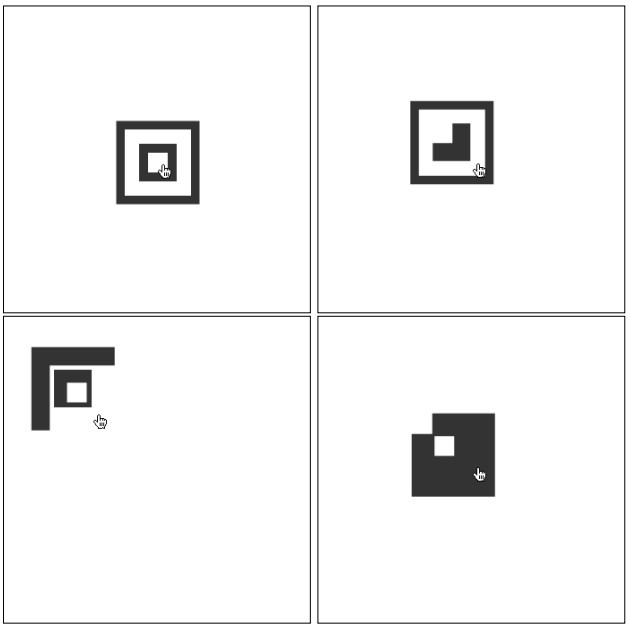
As Actionscript doesn't support pointers, a stand-in solution had to be created in the form of a Reference class. This would be required anyhow as to support relative mapping requires previous values to be tracked, not only for input, but for any value that drives another. To support more exploration with latency, References to variables track values continuously, allowing for access to values or changes in values at arbitrarily previous times.

While developing a library to support these abstract relationships, I made a conscious choice to restrict myself to 'pre-existing' form; while the system should theoretically be used with any quantifiable value, I was not concerned with developing any methods for generating form. (Though I would investigate this possibility later in my *drawing* studies.)

To compensate I've found working with a set number of primitive, overlapping, black and white forms to at least begin to allow for the construction of the kinds of novelly transformable visual forms created by computationally based parametric structures (see *relationships 05*.)

Might be a good place to show materials from intro motion classes...

relationships 05



As the library could support support relationships between any property, I realized I could tie qualities to a change in 'time', which could prove very valuable. The idea of interaction through triggered animation is prevalent in interactive artifacts; For example, one of the building blocks in Flash is the "Movie Clip", a user created, self contained animation that can then be stopped, started, or sent to a specific frames.

There also exists various code libraries (Like greensock.com's TweenLite) that allow for specific computationally driven animation by making a property of some object a function of time. Where-as hand made timeline animation allows for individuals to craft key-framed animations, this approach is often used to create animations "on the fly", in response to some input, and modified based on some variables in the system. For example, an animation could be coded that moves a shape to the cursor each time the user clicks. Each animation would take the same amount of time and utilize the same kind of slow/fast/slow movement regardless of where the box begins or ends.

So far my language of manipulation has been concerned with describing how some property of some object will respond to input. However the language could just as well describe a manipulation that has occurred by noting the beginning and end values of the property, the duration of the change, and the value space through which it changed. Sans input, this is also the same information required to describe a coded animation. In this sense, a coded animation of the kind mentioned could be described as a kind of pre-recorded or pre-made manipulation being played back. Alternately it could be described as a manipulation with time as the actor, and an easing equation used for the mapping. This is discussed later in the section on the drawing studies.

With both user input and time able to drive change it's possible or a manipulation to be constrained by what at least seems to be another actor. *relationships 08* is one experimentation where both time and user input can affect a series of boxes. In it the change in position of the right most block is driven directly by the change in time. The mapping is based on sine creating periodic motion. This box's position is in turn mapped to other boxes with increasingly offset latency and increasing reduction, until the let most block which is entirely unaffected. The left most block is then directly manipulable. It's position though is also used as a driver for other blocks. The result is similar to a jumping rope held by two people at opposite ends.

This kind of set up, where a value is dependent on two different other values, introduces some ambiguity into how a driven object should transform however. One way of handling the situation is to average out the input – perhaps even allowing for some kind of weighting of the various values. Complications arise though when a series of relationships creates a feedback loop. In which case a change may be amplified ad infinitum. The easiest way to achieve this kind of feedback is in attempt to "tie together" two elements so that any manipulation of one effects the other:

Δ box ^{1} position -> Δ box ^{2} position
Δ box ^{2} position -> Δ box ^{1} position

In such a case a manipulation of box1's position (Δ mouse position **->** Δ box ^{1} position) should affect box ^{2} and vice versa. This becomes more complex if a property were to be informally 'driven' by others; such as the distance between the two boxes. Ideally the result should be as intuitive as possible. In this case the expected result is likely that the two move

together as one no matter which is manipulated. In this kind of situation it may be most accurate to make a distinction between mono-directional relationships and bi-directional relationships, or perhaps establish a kind of 'directional' quality of relationships that can be manipulated. However, I'd like to favor a system with more emergent qualities where simpler building blocks allow for certain behaviors to be described through their arrangement or configuration opposed using a solution that allows for fewer connections but more verbose language. An ideal solution to this and any similar problems should privilege the manner in which we perceive causality and action. For example; there is evidence to suggest that our minds organize events in part by attempting to formulate a singular 'actor' that carries primary responsibility for events (Pinker 219). The topic is far out of scope here; but one of great relevance. My final compromise was to simply allow only one *active* relationship to drive change at a time even if more relationships have been created. Furthermore, change is always propagated through the system from those qualities that are closest to the user input (in this case mouse position). The system itself handles this rule. A user (whether a designer or the user of a built system) shouldn't be able to break it accidentally. (The creation of such a system does includes other, more technical, compromises, though I don't believe them relevant to my thesis.)

Multiples
Distance from input.

The functional result of these relationships is similar to functionality offered (as supporting tools) in various animation suites. Adobe's *Aftereffects* animation and video editing software provides a "pick whip" tool on each layer that allows one to be "parented" to another so that transformations of the parent layer affect the child as if it were contained in the parent's co-ordinate system. This tool can also be used to create absolute relationships between different properties of a layer so that one quality might be adjusted through the manipulation of another. Maxon's *Cinema 4D* (for 3D modeling and animation suite) supports the kind of mono–directional delta linking through the options "set driver" and "set driven" accessed by a right click on any numeric property. As useful as these tools are for animation, I believe an application for creating interactive artifacts built primarily on these kinds of tools would be an even greater boon.

Even rigs have clicks and drags though...

So far, these relationships are useful for what might be termed "rigs" in animation — controls, or an interface, for the manipulation of a complex or multi faceted form. Ideally the same language of manipulation and any tool built from it should be able to describe, at minimum, basic responses like 'roll overs' and 'clicks'. This is possible so long as the forms have quantifiable properties like 'touched' or 'pressed' that can be used to drive other qualities. For example; a box's 'touched-ness' could be set to drive a quality of the mapping between the box and the mouse. Furthermore, we might grant that any mapping has a property 'suspend' so that it may be deactivated or re-activated at will.

```
mouse position -> ` box position.
box touched ->'' (-> ` suspend )
```

Two examples of simple interaction built from similar basic clauses exist in the studies *clauses_01* and *clauses_02*.) These kinds of values do not exist as such in actionscript currently, so any library would need to make

accommodations for them. (This isn't to say that actionscript or other ECMAscript based languages don't allow for designed interactivity of the kind described here.)

Programming language design.. far out of scope

The ability to map a binary value like "pressed" vs. "not pressed" to an otherwise continuous value like the multiplier of a mapping function should also work in reverse. In *relationships 12* a set of boxes is mapped to the mouse (after clicking), but the otherwise continuous value of the mouse position is restricted to multiples of the boxes height.

This series of studies begins to end, and the initial form of the manipulus library set with *relationships 13* where five boxes are each manipulable in a different way based on the variables described, creating a different interactive feel for each one. This version of the library had numerous quirks and ambiguities, but would allow for future experimentation.

An ideal execution would be best informed by an investigation of programming language design which is beyond the scope of this paper, and at this point the exact language may be less than ideal. An ideal solution would require a custom parser for a specifically and carefully designed syntax.

For reference, the general form of these mapping statements as supported by the manipulus library is such:

```
Mapper.map(    drivingEntity          : Object,
               drivingProperty        : String,
               drivenEntity           : Object,
               drivenProperty         : String,
               mappingFunction        : Function (or String), // optional
               latency/offset         : int                   // optional
       ): Mapping
```

Mapping

The returned 'Mapping' is an object with qualities pertinent to the mapping like 'active'.

Example use:

```
var mapping     = Mapper.map(stage,'mouseX', box, 'x');
var metaMapping = Mapper.map(box,'pressed',mapping,'active');
```

//
//
//: Drawing and the Manipulation of "Stuff."

After establishing a basic library to work with, I took a small detour to explore the intersection of two issues. The first – that presented by dynamic form – I have discussed earlier. The second concerns a criteria for evaluating interaction that comes from game design: Meaningful play. The investigations helped me refine my description of mapping and expose some perceptual and computational issues concerning the manipulation of multiples. The results begin to move outside the scope of this thesis, but may be compelling avenues for future investigation.

As mentioned in the previous section, the terms of manipulation allows for the qualities of a potential manipulations to change. This allows for past input to affect the results of future input, a key quality in artifacts that are “more interactive”. It also might be considered an abstract way of describing Salen and Zimmerman’s more intuitive game design concept of “Meaningful Play” (61). In these terms gameplay is described as a series of choices, where interesting games provide interesting choices. Interesting choices are defined by having both apparent and immediate consequences but potentially more ambiguous future consequences. This is opposed to “saddle points” (241) where the ideal choice is obvious and the other extreme, where choices produce completely unexpected results. Salen and Zimmerman argue that these choices may be highly cognitive such as deciding what a player’s character might say to another, or very low level, such as positioning a spaceship in order to avoid oncoming fire. In short, choices should be significant in that they have consequences on decisions.

A low level language that describes interactivity as an emergent phenomena — whether mine or some other — should facilitate creating artifacts that support meaningful play – potential manipulations that are distinct, but contingent on, previous manipulations. The mechanics of this may be beyond the scope of this paper. However, I fid find one way to investigate this phenomena.

The description of meaningful play above implies a very literal change in a form’s manipulable qualities, for example: what was turnable is now movable, what was freely draggable now has constraints, what responded quickly now responds sluggishly, etc. It occurred to me though that the act of drawing might involve no change in the manipulable qualities of an entity (a pen) while the use of it still becomes contingent on previous manipulations (previous marks). The manipulation involved in making the first mark on a page is physically the same as making the last mark, but the perception of the two actions may be distinct due to the visual context of the mark making. In this case the manipulation is changing for subjective reason.[6]

Alone, this is similar to the potential emergent interactivity I discussed before; where continual interaction and familiarity change the subjective understanding of the gestures that have objectively remained the same. In the latency studies, different movements often created different kinetic qualities, in the case of drawing, different manipulations create different forms.

More importantly, drawing to the screen — albeit repeatedly at superhuman speeds – is the root of computer generated form. Couching it terms of the ‘manipulation’ of ‘drawing tools’ may make it more approachable. In fact, this idea is similar, if not the same as, the drawing done in LOGO via the command controlled movement of the ‘turtle’.

Source wikipedia

LOGO’s turtle leaving behind a line as it travels.

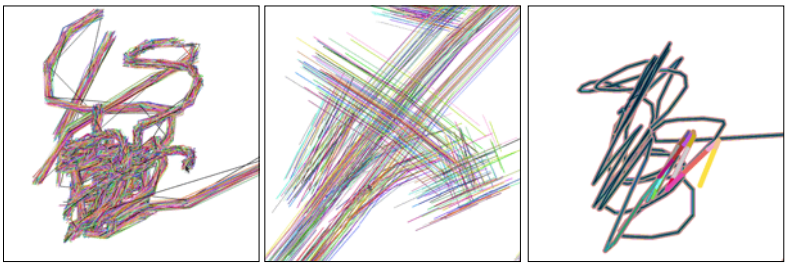
---

[6] It is also very easy to make an imaginative leap to ideas where previous marks have very real effects on the ability to manipulate the pen. Games like ‘Snake’ and ‘Tron’ come to mind.

This idea of manipulating mark-making 'things' may help bridge the theoretical gap between the manipulation of existing form and the generation of new form.

While the act of drawing itself may be no more compelling than the interest a person has in drawing, it should be possible to create novel and compelling results through variation in the mapping and delay of a collection of pens. The following *drawing studies* remind me of the childhood activity of drawing with a fist full of crayons. In this case however, the crayons are spread out in time in addition to space.

drawing 01**, **drawing 02**, and **drawing 03



The manipulus at this point kept constant track of the change in any value it was watching, which effectively gave me access to things like the velocity of the mouse x and mouse y. While I was trying to avoid changing manipulable properties in response to previous actions, I did start utilizing these time based variables to change the pens' mappings. For example, in drawing 02, whether or not a pen draws is dependent on the velocity of a cursor; lines only being drawn with fast movements. At the time I used a series of nested conditional statements in a loop, but it should be possible to update the library in order to make statements like:

```
mouse velocity / 10 -> pen isDrawing
```

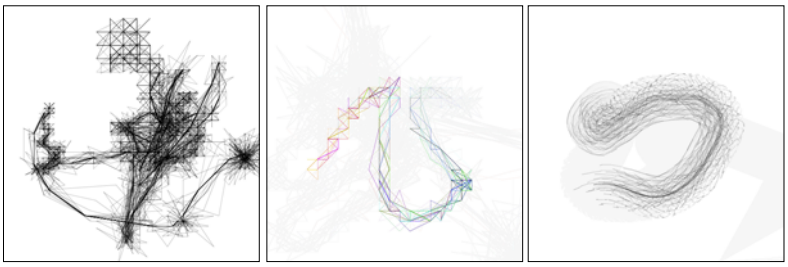
In drawing 04, the velocity of the cursor drives the amount of noise in the mapping of the pens' position; moving the cursor quickly results in lines being draw erratically. Again, the exploration here was based on more traditional coding techniques, but the behavior may be expressed conceptually as such:

```
mouse position -> pen position
mouse position -> (->) mapping noise
```

The experiential result is more similar to manipulating a novel, reactive, form than to the experience of drawing.

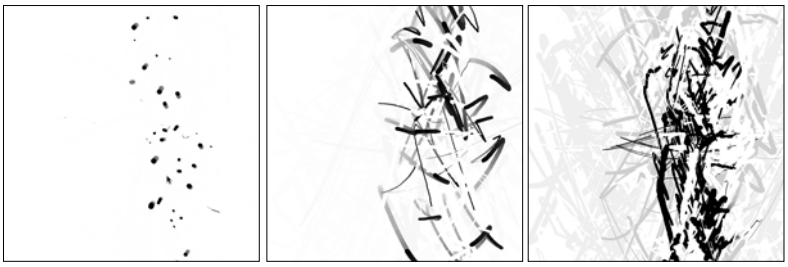
If the older, previous marks, were removed in some fashion, the user left with just the immediate marks, the results would appear less as a drawing, and more as 'reactive' or computationally generated form. In reality, generative form, even when at times when it appears static, involves repeatedly drawing and clearing an image space. Combing this erasing — even if behind the scenes — with the delayed and alternatively mapped pen manipulation creates a variety of manipulable form. How this erasing makes sense with a language of manipulation is at this point unresolved. The visual results were compelling however, so some samples are included below.

drawing 05a**, **drawing 06**, and **drawing 07



The final two studies allow the amount of 'erasing' to be controlled by the user. The more successful one is presented below.

*Samples from **drawing 10***



Beyond the overlap with generative form, these studies exposed for me two problems (or opportunities). The first involving the manipulation of mapping, the second an issue of multiples.

In these cases making the mapping of a manipulation itself manipulable presented both a strong technique of creating both variation and pleasant kinetic qualities, but also some technical hurtles. In drawing 04 it'd be most accurate (though perhaps more difficult) to say that the cursor velocity was driving a noise property *of the mapping.* (The mapping itself a property of the manipulation). To say that the mapping of any relationship has a "noise" property is problematic however. In fact, to say that all mappings have a common set of properties is impossible.

On the computer side, the mapping of one value to another is expressed as a mathematical function, and such functions can take an endless variety of forms with a endless variety of potential properties.

The most generic mapping function is linear. It relates one input value to one output value. i.e. If the cursor's x position is ever 100, the mapped value is guaranteed to be 200; if the cursors's change in x position was 10, the change in the mapped value is guaranteed to be 10.
Linear equations are expressed mathematically as such:

f(x) = x;

A simple variation on this mapping would be to change the output value by multiplying it or adding to it:

f(x) = x+1;
f(x) = x/10;

Note that one form has an offset while the other has a multiplier. The forms are distinct. The range of potential linear equations is endless:

f(x) = 2*x+1;
f(x) = (x*x) + 2*x + 2;
f(x) = 3*x*x+2(x*x) + 3*x + 3;
...

This problem becomes more difficult (or interesting) when additional 'free' parameters are allowed to enter the equation. These equations are known as parametric equations as they describe a *range* of linear functions, just like parametric forms describe a range of form.

The set of equations including an offset, where the offset may be any number:

f(x,n) = x + n;

The set of equations including a multiplier, where the multiplier may be any number:

f(x,n) = x * n;

These kinds of multi-parameter equations are useful in that they may be used to describe behavior such as: the pen's x position follows the mouse's x position, but becomes more erratic as the mouse y increases:

Create example

f(x,n) = x * random(n);

Parametric equations are often used in animation programs or code libraries for the use of describing a *kind of movement* ("start slow, then speed up") while leaving the beginning, end, and duration of the movement variable. A simple parametric equation that produces a constant change over time takes the form of:

```
current value = start value + (end value - start value)*(elapsed
time / duration);
```

While a single parametric equation can be transformed into endless forms of linear equations (in order to describe movements of various durations with various starting and end locations), different parametric equations still describe distinct sets of linear equations Two different parametric equations will describe two distinct kinds of movements while the movement's duration, starting and end points are the same. The animation created by the above equation gives the object a constant velocity and is generally considered stiff, unnatural, or robotic. In contrast, other types of parametric equations can be used to describe an animation that has the kind of 'slow-in, slow-out' movement advocated by animation pioneers Ollie Johnston and Frank Thomas. These different equations can not be transformed into one another however. By extension, animation libraries typically require that a specific parametric equation be specified when an animation is created, e.g. Quadratic, Quartic, Exponential, Bounce, Elastic, etc.

http://www.robertpenner.com/easing/penner_chapter7_tweening.pdf

Edge animation drop downs as example.

The variety and non-overlapping nature of these equations that describe the computational aspects of a mapping prevents me from providing one set of convenient, manipulable, 'properties' to manipulate. There are compromises however.

One practical approach is to utilize a default parametric function to use, and allow for a user to input more complex parametric functions if they wish (and are inclined). This was my eventual approach. It allows for simple mappings to be expressed simply. In the following Actionscript example the pen's x position is always equal to the mouse's x position divided by 100.

```
Mapping.map(stage,"mouseX",pen,"x","/100");
```

As mentioned, more complex equations may be also be used, but doing so requires the programmer/designer to specify them explicitly. In the next example the pen's x position is always equal to the square of the mouse's x position plus the mouse's y position.

```
function mapping(input){ input*input + offset};
mapping.offset = 100;

Mapping.map(stage,"mouseX",pen,"x",mapping);
Mapping.map(stage,"mouseY",mapping,"offset");
```

It may be that some equations are more desirable by designers than others. A better solution could come from getting the library into the hands of other designer/programers.

In addition to mapping, the drawing studies helped foreground a perceptual distinction between the manipulation a thing, the manipulation of things, and the manipulation of 'stuff'. Along with this, a personal desire to more accurately describe relationships between one-and-many, and many-and-many.

In the early Latency studies it became quickly apparent that while it was possible to focus on a directly manipulable entity in crowd of similarly behaving items, focus often seemed drawn to the artifact as a whole. Here with the drawing studies In the absence of any controlled, visual, entities this phenomena is more pronounced. Perceptually, it appears as the manipulation of 'stuff'. Presently, my ideas focus on the manipulation of a 'thing', or perhaps the manipulation of multiple 'things' where the attention moves rapidly from one to another. The use of a collection of things however gives the artifact as a whole a unified gestalt, but also the sense of manipulating something amorphous and indistinct. Research in linguistics has found that individuals make a particular distinction between "collections" of items and amorphous "stuff" (Pinker, 167–174). The perceptual shift from the manipulation of a thing, to things, to stuff is likely not continuous, and could be the focus of further exploration. Regardless, language of manipulation (and its technical implementation) should support the creation of mappings between many things. Even more useful would be to support the manipulation of the relationships between 'things'. Just as a mapping is a modifiable relationship between manipulator and manipulee, it would be useful if spatial relationships (like the distance between two items) could themselves be the target of a manipulation.

In computation the 'loop' is the de-facto tool for changing large quantities of data. The ability to have a computer do such repetitive tasks is arguably their most powerful and distinct qualities as a tool or medium. When the level of repetition goes beyond a typical human's mental capacity — become an abstract computational concept that must be learned. Applying it, conceptually, in the creation and modification of experiential phenomena is a sophisticated skill requiring a great deal of practice.

For some operations it may be more ideal if the language could provide a more intuitive method of dealing with quantities. Some design decisions, like mapping the mouse position to the rotation of 100 squares, should be accomplished with brevity and elegance. This requires an elegant and sophisticated method for articulating and resolving selections, e.g. "All the squares", "All the black squares", "All the squares to the left.". Alternatively, an ability to label, create, or set aside arbitrary selections for future manipulations may also be useful.

Both approaches are utilized in common web design practices. Firstly, HTML tags can have a singular identifier, "id", any number of group identifiers "class", and a distinct structural relationship. (Content elements are typically contained within another element.) In turn, the

language of Cascading Style Sheets (or CSS) can be used to define visual styles for many elements with brief, simple, but powerful statements. The following CSS code will cause any number of header elements in a document to be displayed in a bold typeface:

```
h1{ font-weight:bold; }
```

CSS also supports more advanced selection statements such as “The first child *element* of *all* of some type of element”. This allows for stylistic treatments like making the first line of a paragraph italic without explicitly defining those particular items. These selections may also be malleable; if the column width of the paragraph changes, the style rules will apply to those words *currently* in the first line, opposed to those words that were *initially* in the first line.

Using these selection abilities, Javascript libraries like jQuery can change visuals rules in response to actions taken by the user. For example, clicking an item in a list changes the visibility of a sublist; the net result being a basic drop down menu. jQuery can modify style properties of a single item or a collection of similar items without a change in syntax. The following code will change something to red whether that something is one element or a collection of one hundred elements.

```
$(“.header”).css(“color”,”red”);
```

There are other technologies that include similar abilities such as the ECMAScript extension E4X for working with XML documents and the relatively complex but powerful Regular Expression syntax for use in finding selections in a text such as “any number that is preceded by a bullet point”.

The power of sophisticated selection abilities are seen in user interfaces too. Many existing tools, particularly 3D modeling applications, provide a variety of approaches to dealing with large quantities of manipulable items, usually in respect to their geometric relationships. Modelers are often provided with ways of selecting continuous loops of polygon edges or the creation of weighted selections of points where specific relationships are mapped more or less tightly based on the a specific point’s distance from an initial click.

Eventually I implemented a rudimentary way of dealing with selections in the manipulus library in the form of Sets. Sets are intended to eventually act like a jQuery collection in order to allow for such statements as:

```
mouse x -> a set of shapes’ x position.
```

Currently, the Sets implementation includes a ‘selected’ property that reflects the item that will be manipulated when using the set. This has the added benefit of creating a relationship to a place holder and allowing for the target element of a manipulation to itself be modified.

Include Example

include example

Furthermore a Set provides several properties that describe the relationships between forms, such as “distance” and “angleBetween”. These can be used to easily make one object point to another.

```
var set = new Set(arrow1, arrow2);
Mapper.map( set, 'angleBetween', arrow1, 'rotation');
```

Further research into both the ways in which we perceive collections of entities and their spatial relationships, and computational methods of describing groups would be useful to discover both what is most useful, and where common sense descriptions translate ambiguously into computation. For example, the desire to change the rotation of a set of boxes might translate to either the rotation of each box, or the transformation of the boxes as if it were a single object.

//
//
//: Application

With a basic working library to handle the computational description of manipulations it stands that developing a GUI for such statements is possible. Here begins a new and relatively sophisticated design challenge. The result would be a system for the direct manipulation of manipulation, and by extension, the direct manipulation of interactivity. While the creation of a complete application is of a scope worthy of an entirely new project, I found that continually considering how to represent my ideas about manipulation in a visual, manipulable form helped me identify how I’d like to ‘get my hands on’ the structures I was proposing, and in what ways. The need to touch on relationships of multiples was one such insight reinforced by this exercise. An issue with the propagation of user input through a chained system is what initially forced me to reevaluate the how reciprocal relationships were articulated both naturally and in code. It helped me understand how ambiguous certain statements could be such as “the two boxes affect one another.”

The initial problems in building such a system are visual communication in nature; simply representing the relationships in an intelligible manner and in a way distinct from the manipulable form itself. Next, while it may be easy to form connections between different elements, any “interaction” is predicated on some kind of relationship between the artifact’s forms and the input. A designer would then require some self-reflexive ability to point to the point-of-input — the input which was being used to manipulate the tool in the first place. While an interface that is completely based on direct-manipulation sans hot keys or other accoutrements would be ideal, this introspective need to “refer to oneself” using the artifact at hand interferes with the potential for a real-time all-the-time editor. There will inevitably arise a moment when a user will need to either suspend the artifact, or step outside of it. Having two distinct modes is unavoidable.

In other tools this is handled by entering and exiting ‘editing’ mode or ‘locking’ the tool so that changes may not be made. The design challenge is then to both minimize the interruption so that a continuous

work flow can be maintained yet make the distinction between the two modes clear. Ideally the tool's cursor could become detached from some kind of meta-cursor when entering the edit mode. This requires decoupling the default cursor from mouse or trackpad input, a feature otherwise known as "mouse lock", which is currently unavailable at the time of writing in the development environment I began in (Flash).

One difficulty I came across was circular references. The moment I was able to select an item I wanted to select several and create relationships between them. This requires either a mapping from each property to each other property or a mapping from one to the next and from the last to the first. Either setup results in a situation where changes to one propagate through the chain of causality and back to the originator, only to continue through again, infinitely. The result is that a disturbance in one value causes the entire series of linked values to jump to infinity. The solution was to prevent more than one mapping to be in effect at a time. While it may be possible to declare that some property is to be driven by both mouse input and other properties, the moment that mouse movement influences the value, that value will be modified by any other changes in the system. Changes from outside of the system thus have top priority, overriding any potential changes from inside.

Future version may be more expressive if this was not the only option for the resolution of multiple inputs. The problem also exists in animation software like Cinema 4D where users can take advantage of both key-framed and physics based animation (among other techniques for creating change over time). In these systems the priority and even weighting given to various inputs can be set explicitly by the user. This might be useful functionality in the future.

(wip Prototype: Video**)**

What about relationships? Manipulation of visual relationship. Supporting Perceptual Shifts. vs. avoiding modes... Gap tool in In design

//: A Revised Definition and Recommendations

While the initial definitions appear to provide some ability to make meaningful modifications to various interactive artifacts, it's apparent that it could use much refinement.

What I referred to as latency would be better expanded and described as the "temporal space" of a manipulation; the period of time over which a manipulation happens. This space might be described in terms of duration; the time between the moment an entity's change is understood to be the effect of input and when the the desired change is complete. Note that this moment of understanding may be subjective. A individual new to a complex game with a great deal happening on screen may not recognize that their inputs are responsible for various immediate changes on screen. Alternately, in a situation where objectively observable screen changes are not synchronized with input, it may be possible for a user to understand the visual 'silence' as a meaningful and understandable response to their input, thus constituting the beginning of a manipulation from a perceptual point of view. This can be seen specifically in iOS devices that employ hold-to-active buttons.

In these cases there seems to be an additional dimension regarding input. Several manipulations with the same “duration” may each require either continual input, momentary input with an anticipated but delayed result, or a momentary input with potentially variable results that require continual attention. Further investigation may provide a way to distinguish between where a manipulation may “bend”, and where it may deviate so much as to qualify as a distinctly new manipulation. For example. If the amount of movement required to move a square changed gradually as it was moved, it might be perceived as a continuous but changing manipulation, where as if it changed drastically and instantaneously it might be completely interruptive and necessitate some kind of mental re-evaluation.

The “temporal space” of a manipulation could be further divided to include a ‘middle’ and more importantly an ‘end’. Steve Swink in “Game Feel” describes the response to direct manipulation in acoustic terms: Attack, sustain, and decay (121). I have not experimented with these divisions, but they may be very helpful categorizations, particularly in regard to a where a manipulation’s end is marked. Previously I mentioned how a the objective results of a manipulation may be ‘good enough’ to subjectively qualify as complete. Here the term Tolerance makes sense and I’ll define as: The acceptable variance in the ending value for a manipulation to be perceived as complete.

Mapping also leaves some problems. The relationship between input and output is qualified in very mathematical terms that may not have intuitive equivalents. I’ll described this instead as a ‘value range’. As such it may make sense to describe this range of values as having potential variation much like the there may be variation in the end value of a manipulation. As such Tolerance could also be defined as: The acceptable variance in the value space before a manipulation breaks down, is discarded, or revised in some fashion.

Avenues for further research are numerous. I believe the study of the Perception of Action in psychology could yield some great insights, and would be an extremely valuable step in understanding what may be the universal aesthetic qualities of interaction. Philosophically, I understand that phenomenology and the writings of Maurice Merleau-Ponty are the source of much body-centric writing in HCI and IxD. These theories I have only tertiary knowledge of and could not integrate into this thesis. Studies in programming language design could help could help shape a language of manipulation into something that is both approachable and computationally powerful. Critically, it might be interesting to compare my definition of interaction to game designer Jonathan Blow’s idea of orthogonality in game mechanics (part of a larger aesthetic ideal of game design), or even to use my definition in evaluating existing artifacts. The interrelationship of an artifact’s kinetic and interactive qualities and their role in its dynamic gestalt could needs explored. A more refined application for the creation of interactive artifacts without the use of programming would be useful. Lastly, artistic experimentation could focus on applying a language of manipulations to semantically and culturally rich elements.

Needs to be MLA not chicago

//
//
//: Works Cited

Brathwaite, Brenda. “Built on a Foundation of Code – Game Edu Rant «Applied Game Design”, n.d. http://bbrathwaite.wordpress.com/2011/03/01/built-on-a-foundation-of-code-game-edu-rant/.

Brooks, Frederick P. *The Mythical Man-month: Essays on Software Engineering*. Anniversary ed. Reading, Mass.: Addison-Wesley Pub. Co., 1995.

Buxton, William. *Sketching User Experience: Getting the Design Right and the Right Design*. San Francisco, CA: Morgan Kaufmann, 2007. http://www.loc.gov/catdir/toc/ecip074/2006036416.html.

Cooper, Alan. *The Inmates Are Running the Asylum*. Indianapolis, IN: Sams, 1999.

Crawford, Chris. *The Art of Interactive Design: a Euphonious and Illuminating Guide to Building Successful Software, June 2002*. San Francisco: No Starch Press, 2002. http://www.loc.gov/catdir/enhancements/fy0716/2002001562-d.html.

Cypher, Allen, and Daniel Conrad Halbert. *Watch What I Do: Programming by Demonstration*. Cambridge, Mass.: MIT Press, 1993.

Dourish, Paul. *Where the Action Is: The Foundations of Embodied Interaction*. Cambridge, Mass.: MIT Press, 2001. http://www.loc.gov/catdir/toc/fy036/2001030443.html.

Gingold. *Miniature Gardens & Magic Crayons*, n.d.

Gingold, Chaim. “Catastrophic Prototyping and Other Stories | Levitylab”, n.d. http://www.levitylab.com/blog/2011/01/catastrophic-prototyping-and-other-stories/.

Lakoff, George, and Mark Johnson. *Metaphors We Live By*. Chicago: University of Chicago Press, 2003.

Lopes, Dominic. *A Philosophy of Computer Art*. London: Routledge, 2010. http://www.loc.gov/catdir/enhancements/fy1106/2009007750-b.html.

Löwgren. *Five Things I Believe About the Aesthetics of Interaction Design*, n.d.

Löwgren, Jonas, and Erik Stolterman. *Thoughtful Interaction Design: a Design Perspective on Information Technology*. Cambridge, Mass.: MIT Press, 2004. http://www.loc.gov/catdir/toc/fy052/2004049891.html.

McCullough, Malcolm. *Abstracting Craft: The Practiced Digital Hand*. Reprint. The MIT Press, 1998.

Mccullough, Malcolm. *Digital Ground: Architecture, Pervasive Computing, and Environmental Knowing*. The MIT Press, 2005.

Norman, Donald A. *Emotional Design: Why We Love (or Hate) Everyday Things*. New York: Basic Books, 2004. http://www.loc.gov/catdir/toc/ecip043/2003010123.html.

———. *The Invisible Computer: Why Good Products Can Fail, the Personal Computer Is so Complex, and Information Appliances Are the Solution*. Cambridge, Mass.: MIT Press, 1998.

Norman, Donald. *The Design of Everyday Things*. Basic Books, 2002.

Raskin, Jef. *The Humane Interface: New Directions for Designing Interactive Systems*. Reading, Mass.: Addison Wesley, 2000.

Reas, Casey, and Chandler McWilliams. *Form+code in Design, Art, and Architecture*. 1st ed. Design Briefs. New York: Princeton Architectural Press, 2010.

Saffer, Dan. *Designing for Interaction: Creating Innovative Applications and Devices*. 2nd ed. Voices That Matter. Berkeley, CA: New Riders, 2010.

Salen, Katie, and Eric Zimmerman. *Rules of Play: Game Design Fundamentals*. Cambridge, Mass.: MIT Press, 2003.

Sharp, Helen, Yvonne Rogers, and Jenny Preece. *Interaction Design: Beyond Human-computer Interaction*. 2nd ed. Chichester: Wiley, 2007. http://www.loc.gov/catdir/toc/ecip071/2006030649.html.

Shneiderman, Ben. *Direct Manipulation: A Step Beyond Programming Languages*, n.d.

Snyder, Carolyn. *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. San Diego, CA: Morgan Kaufmann Pub., 2003. http://www.loc.gov/catdir/description/els031/2002115472.html.

Svan\aes, Dag. *Understanding Interactivity: Steps to a Phenomenology of Human-computer Ineraction*. Trondheim: Norges teknisk-naturvitenskapelige universitet, Institutt for datateknikk og informasjonsvitenskap, 2000. http://www.loc.gov/catdir/toc/fy0609/2001319087.html.

Ware, Colin. *Visual Thinking: For Design (Morgan Kaufmann Series in Interactive Technologies)*. Illustrated. Morgan Kaufmann, 2008.

....

//
//
//

s c r a t c h

//
//
//

// the relationships series of studies develops during this time.

//
(Resemblance to dancing)

(clauses / other developments / development of a pre-existing form with qualities 'touched' and 'pressed' [perhaps any form should posses these qualities based on subjective perceptions of where a shape contains a point, though this might be based on functional relationships...])

(dealing with multiples :/)

//

The studies *clauses 01* and *clauses 02* also provide examples. This time using a formulaic property "touched" to

//
//
//

Physics vs. Manipulation
Ability to assimilate to non-representative simulations.

Studies:

// fidelity / simultaneous contrast xx
// token chain 6.30.10, 12.21.10? xx
// latency studies : 6.30.10 xx
latency as variable.

// finger studies. 7.1.10? 6.30.10? xx
Different 'latencies' for different properties.

// tokens 7.30.10 ? 7.1.10? ??
Same manipulable qualities but with different **constraints**

// drag box: 12.18.10 ***~~***

Study: Dragging x effects how y will react when dragged.

Study: box height -> mapping multiplier.

Binary Study: Immediate change in place of gradual change.
Mapping ends at a certain point. Constraint.

Binary Study: Immediate change.
Quality changes at a certain point.

// ring box 12.18.10 **xx**

// parametric forms

Different computational descriptions allow for different manipulations. Some descriptions may overlap at a 'point'. Some may overlap continuously...

Multiple boxes?
Multiple ways of manipulating the same box.
(lots of things support this, alternately referred to as 'modes' or 'metaphors' 'structures' or 'ontologies'. Multiple modes allow for errors, particularly when its ambiguous. Several approaches exist to avoid this: 'tools': selecting a tool switches between 'ontologies'. Eliminate modes all-together. This is impossible, in the case of a layout program where text may be manipulated as language but also as visual form.

this tension is everywhere.

// relationships 1.6.11 **...**

relationships 12 : sampled possibility space
relationships 13 : 5 boxes, five different interactive qualities
relationships 14 : boxes and width

// fidelity : 1.7.11 **xx**
// boxes : 4.28.11 **xx**
// dancing boxes 4.9.11
// clauses 6.26.11 **~~**
// drawing 7.3.11 **~~**

// CONCENS / THOUGHTS

More game design
Interactive vs Kinetic qualities.
Continual change vs state machines.

The point is not to support the creation of useful solutions, but solutions that are engaging, learnable, communicative, *and* useful.