FIGURES NEED CAPTIONS

Introduction:

Deciding who to vote for is hard. The deliberations and conversations that go into choosing who best represents one's interests is an important and time-consuming task, one that might be argued to be the very backbone of a democracy. Understandably, many may assume that the subsequent task of correctly voting for one's chosen candidate is comparatively easy and straightforward. Surely once a voter gets the ballot and can mark whoever they please, the hard part is over.

Hmm

For the most part, they would be right. When ballots are designed well, any error voters make are not systematic and will not help or hurt any candidate. When they are designed poorly, they may lead to systematic voting errors, but perhaps this happens rarely or does not matter on a large scale. In the general case, people tend to make relatively few mistakes when marking a ballot with who they want to vote for.

However, in closely contested elections it is not the general case that is important. There have been numerous elections in the past 20 years that have been documented as having been decided by systematic voting errors caused by bad ballot design. While election interference by hacking is a far more flashy and obvious risk, there at least has never been evidence that this has swung an election, unlike for bad ballot design. Ironically, the fear of hacking has led to a return to paper ballots, which with their profusion of races packed onto small sheets of paper makes ballot design even more important.

ALSO, MOST ERRORS ARE WRONG CHOICE

DEFINE

This problem of designing ballots that will not cause people to systematically under or over vote is even more challenging than it first appears because of the sheer number of counties in the United States, each of which designs their ballots differently and each of which have hundreds of different iterations of ballots for each precinct they are responsible for. It is thus impossible to run a traditional usability study on every iteration of every ballot in every race in the country. It is simply a matter of scale.

UP TO HERE IS GOOD BUT TOO LONG

One possible solution to this problem is a piece of software that could automatically check an arbitrary ballot for common ~~voting errors~~ DESIGN ERRORS. However, since the task is to decide if humans will make a mistake, it is initially difficult to imagine building an automatic system to do such a task. Here is where ACT-R, a cognitive architecture purpose built to simulate human cognitive tasks, comes in. In order to ensure that every mistake was predictable, any mistakes a human would make on a certain ballot would have to be made by the system as well

This is not an original idea. Green (2010) built an ACT-R model that could make the same mistake voters did in a famous ballot caused systematic error. However, these and other attempts have focused on replicating one error behavior. A system that could be used by election officials to identify bad ballots would have to be able to predict all historical voting errors, as well as any new ones.

Thus, Wang, Lindstedt, and Byrne (2019) describe the construction of a model that will simulate a vast array of possible voting errors, with an eventual goal of simulating the entire voting space. In their paper they describe the start of building such a model. The model ran in a voting environment called VoteBox. It was a simulated simple electronic ballot, consisting of just a single race per screen with a next button to navigate.

TASK

Nevertheless, within just a simple task was hidden great complexity: Wang describes building and using a total of 40 different voting strategies constructed from differing memory and navigational strategy selections. The voters differing strategies and knowledge led to different rates of error, showing that at least at this small-scale strategy made a difference to whether these simulated voters were able to vote for the candidates they wished for. However, this ~~system~~ made no attempt to vary the structure of the ballot and so did not yet have the predictive power necessary to recognize badly designed ballots.

EFFORT

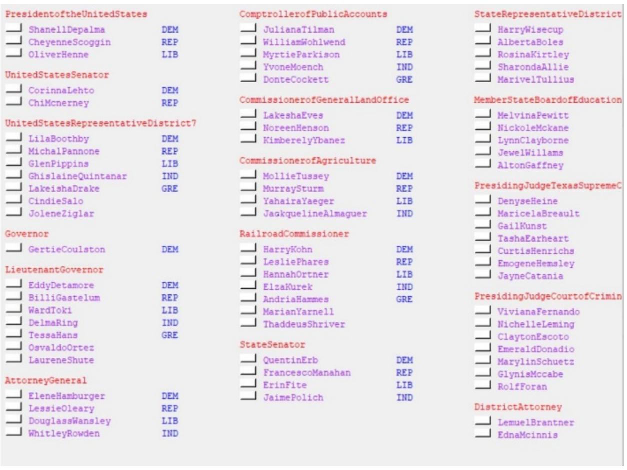
Thus, in this paper, we first describe the extension of this system to handle full simulated paper ballots. Then, we describe the error rates of our new simulated voters on various simulated ballots. This represents a large step towards our end goal of constructing a piece of software to identify bad ballots.

Methods~~:~~

JUST "METHOD" NO S, NO COLON

First, we describe the design of our new system that can vote on simulated paper ballots. The ACT-R environment can interact with a virtual screen. ~~This is how Wang built the initial simulations, except the extension of the task to a full ballot required an entire rewrite of the simulation code~~. ~~Thus, we~~ We built simulated full "paper ballots" for the model. These consist of a virtual screen populated with several columns of races. Each race has a title, a list of candidates and their associated party, and a list of buttons that the model can click to vote for a candidate. This results in a simulation that is not quite the same as a paper ballot, but close enough to cause at least many of the same errors we expect humans to make. Below is one example ballot.

EXPLAIN WHY NOT



PresidentoftheUnitedStates
□ ShanellDepalma DEM
□ CheyenneScoggin REP
□ OliverHenne LIB

UnitedStatesSenator
□ CorinnaLehto DEM
□ ChiMcnerney REP

UnitedStatesRepresentativeDistrict7
□ LilaBoothby DEM
□ MichalPannone REP
□ GlenPippins LIB
□ GhislaineQuintanar IND
□ LakeishaDrake GRE
□ CindieSalo
□ JoleneZiglar

Governor
□ GertieCoulston DEM

LieutenantGovernor
□ EddyDetamore DEM
□ BilliGastelum REP
□ WardToki LIB
□ DelmaRing IND
□ TessaHans GRE
□ OsvaldoOrtez
□ LaureneShute

AttorneyGeneral
□ EleneHamburger DEM
□ LessieOleary REP
□ DouglassWansley LIB
□ WhitleyRowden IND

ComptrollerofPublicAccounts
□ JulianaTilman DEM
□ WilliamWohlwend REP
□ MyrtieParkison LIB
□ YvoneMoench IND
□ DonteCockett GRE

CommissionerofGeneralLandOffice
□ LakeshaEves DEM
□ NoreenHenson REP
□ KimberelyYbanez LIB

CommissionerofAgriculture
□ MollieTussey DEM
□ MurraySturm REP
□ YahairaYaeger LIB
□ JackquelineAlmaguer IND

RailroadCommissioner
□ HarryKohn DEM
□ LesliePhares REP
□ HannahOrtner LIB
□ ElzaKurek IND
□ AndriaHammes GRE
□ MarianYarnell
□ ThaddeusShriver

StateSenator
□ QuentinErb DEM
□ FrancescoManahan REP
□ ErinFite LIB
□ JaimePolich IND

StateRepresentativeDistrict
□ HarryWisecup
□ AlbertaBoles
□ RosinaKirtley
□ SharondaAllie
□ MarivelTullius

MemberStateBoardofEducation
□ MelvinaPewitt
□ NickoleMckane
□ LynnClayborne
□ JewelWillams
□ AltonGaffney

PresidingJudgeTexasSupremeC
□ DenyseHeine
□ MaricelaBreault
□ GailKunst
□ TashaEarheart
□ CurtisHenrichs
□ EmogeneHemsley
□ JayneCatania

PresidingJudgeCourtofCrimin
□ VivianaFernando
□ NichelleLeming
□ ClaytonEscoto
□ EmeraldDonadio
□ MarylinSchuetz
□ GlynisMccabe
□ RolfForan

DistrictAttorney
□ LemuelBrantner
□ EdnaMcinnis

WILL NEED TO BE BIGGER

ON

COLLOR

Note that to help the model navigate, we colored the race header red, the candidates purple, and the parties blue, as ACT-R can make visual location requests based ~~off~~ on color. ~~More on this later~~, but we expect that our model will eventually be able to make these groupings naturally like people do, although this will require an extension of ACT-R itself. Below is one example ballot.

SAVE FOR WHEN THIS IS EXPLAINED

BIT INFORMAL

Thus, our model had a canvas to test on, but we still needed to extend the old system that could only vote on a single race in the VoteBox environment to a new system that could vote on every one of these races. We built this new system with our overarching goal in mind: to simulate as wide an array of voters as possible.

~~Thus,~~ we constructed a modular system that split a simulated voter's strategy into five different pieces: memory, the voter's knowledge of who they were voting for; macronavigation, the

A THING

ODD STRUCTURE

A PROCESS

CONFUSING. HOW IS THIS NAVIGATION?

process of moving from one race to the next; encoding, the process of determining the race, party, and candidate visual groups; micronavigation, the process of determining who to vote for; and clicking, the process of actually clicking on the button corresponding to the chosen candidate. In other words, any specific model does the following in serial order: pick a race, determine where the candidates where, decide which candidate to vote for based off its memory and navigation, and then clicks on ~~the~~ a button. We built a combiner that could dynamically select one strategy from a list of each of these categories and build a voter. Note that Wang's work remains in the micronavigation category, as those strategies fit in nicely as a piece of our new system.

NOT A THING ET AL.

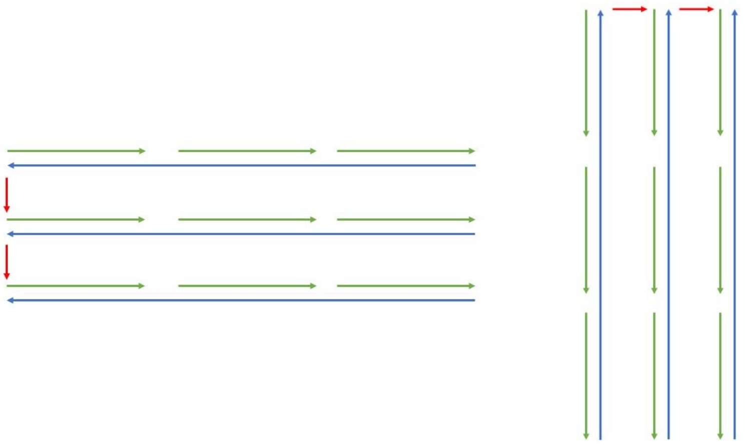
We first built the most obvious strategies for each of these categories. This choice meant that our initial strategies led to no errors. In other words, when we had finished building our first strategy for each case and combined them all into a single simulated voter, the voter made no mistakes. This result was intended. We first wanted to have a *working* voter before we started inducing errors.

AWK

It is timely to note that our model's initial macronavigation strategy was *top to bottom left to right*; that is, the model started in the top left corner and went all the way from the top to the bottom and then went over to the next column to the right and again went top to bottom, and then repeated until finished. This is typically the way ballots are generated and is the most obvious method of macronavigation, and like noted above resulted in no mistaken votes. ?

ONCE ~~Now that~~ we had a working voter, we sought to add an additional macronavigation strategy. WHAT? Although we could choose any of the 40 possible micronavigation strategies Wang had generated, we decided to hold the micronavigation constant with a so called perfect strategy, one that had a perfect memory of who to vote for and so would never vote for the wrong person. Because of the modular nature of our model, this would ensure that any errors were a result of our macronavigation procedures. We want our final product to be able to simulate a vast array of possible voters, but for development it is useful to hold many of the variables constant
.

The first new macronavigation strategy we built was *left to right top to bottom*. This strategy started on the first row of races and then proceeded serially from left to right, continuing through each row from top to bottom. Below is each pattern of movement, where the green arrows represent moving to the next race, the blue arrows a backtrack to the start of a column or row, and the red arrows movement between rows.



PROBABLY CUT THIS

WHOA! EXPLAIN HOW!

Crucially, because the order this new strategy visited races was not matched to the way the races where ordered on the ballot, the strategy *missed races*. Moreover, the races it missed depended on the layout of the races on the ballot. ~~Initially, building this strategy was meant as a test run~~. The error behavior was new, unplanned, exactly the sort of thing that we were going for in building a voter simulation. ~~Instead of moving on and building additional strategies, we decided to explore this entirely novel behavior~~.

COLLOQ

Now that we had a simulated voter making mistakes, we decided to test how these mistakes changed as we modified the ballot. ~~For the first time, we would be able to identify poorly designed ballots by observing how many mistakes our model made on them.~~ We reconstructed the ballot to be dynamically built with an arbitrary number of random races, each consisting of an arbitrary number of candidates. We then varied several spacing variables that guided how the ballot was built each time the model ran. These variables were the spacing between each race, the spacing within each race between candidates, and the spacing between each race between the race header and the candidates Finally, for each one of these possible combinations of variables we ran the model hundreds of times, recording ~~exactly~~ which races the model voted on and which ones it missed.

AWK

WHAT?

↳ BE SPECIFIC

AWK

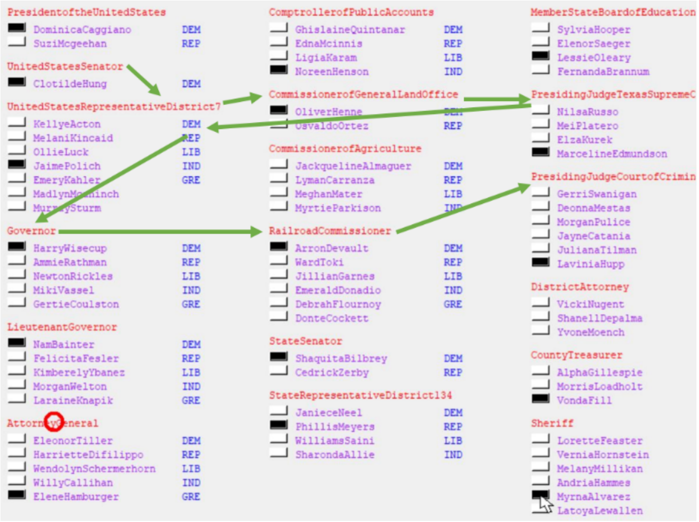
Analyzing this THESE data, as we will do in the next section, has two purposes. The first is to characterize the strategy itself and identify how and where the strategy fails. The second, closely related but not identical to the first, is to characterize good and bad ballot design by seeing which designs lead to more error in the model. To be clear, we will confine our discussion of the model's errors to this left to right, top to bottom strategy, although this serves as a case study for how new strategies built on the same structure as described above will find errors.

Results:

Before we delve into a description of the model's errors, we first show a specific example of one of these errors. Here is a section of the ballot from a typical run of the model, with a segment of the path of the model's attention encoded by the green arrows. When the model reaches the third race down on the left ("UnitedStatesRepresentititveDistrict7") it votes on the race and then proceeds right along the row, selecting and voting on the closest race and repeating until it reaches the right side. The model then returns to the race at the beginning of the row and proceeds to the next row down. Here is where it makes its mistake, because the RailroadCommisioner race is the closest race to Governor, and so as the row is completed and voted on CommisionerofAgriculture is skipped. This mistake is typical for this method of macronavigation and leads to an error rate of about 15.5% across all trials. ~~This statistic is certainly high for a typical voter, but we are not concerned with the typical voter.~~

AWK

WHY DOESN'T IT START HERE?



PresidentoftheUnitedStates
■ DominicaCaggiano DEM
□ SuziMcgeehan REP

UnitedStatesSenator
■ ClotildeHung DEM

UnitedStatesRepresentativeDistrict7
□ KellyeActon DEM
□ MelaniKincaid REP
□ OllieLuck LIB
■ JaimePolich IND
□ EmeryKahler GRE
□ MadlynMcminch
□ MurraySturm

Governor
■ HarryWisecup DEM
□ AmmieRathman REP
□ NewtonRickles LIB
□ MikiVassel IND
□ GertieCoulston GRE

LieutenantGovernor
■ NamBainter DEM
□ FelicitaFesler REP
□ KimberelyYbanez LIB
□ MorganWelton IND
□ LaraineKnapik GRE

AttorneyGeneral
□ EleonorTiller DEM
□ HarrietteDifilippo REP
□ WendolynSchermerhorn LIB
□ WillyCallihan IND
■ EleneHamburger GRE

ComptrollerofPublicAccounts
□ GhislaineQuintanar DEM
□ EdnaMcinnis REP
□ LigiaKaram LIB
■ NoreenHenson IND

CommissionerofGeneralLandOffice
■ OliverHenne DEM
□ OsvaldoOrtez REP

CommissionerofAgriculture
□ JackquelineAlmaguer DEM
□ LymanCarranza REP
□ MeghanMater LIB
□ MyrtieParkison IND

RailroadCommissioner
■ ArronDevault DEM
□ WardToki REP
□ JillianGarnes LIB
□ EmeraldDonadio IND
□ DebrahFlournoy GRE
□ DonteCockett

StateSenator
■ ShaquitaBilbrey DEM
□ CedrickZerby REP

StateRepresentativeDistrict134
□ JanieceNeel DEM
■ PhillisMeyers REP
□ WilliamsSaini LIB
□ SharondaAllie IND

MemberStateBoardofEducation
□ SylviaHooper
□ ElenorSaeger
■ LessieOleary
□ FernandaBrannum

PresidingJudgeTexasSupremeC
□ NilsaRusso
□ MeiPlatero
□ ElzaKurek
■ MarcelineEdmundson

PresidingJudgeCourtofCrimin
□ GerriSwanigan
□ DeonnaMestas
□ MorganPulice
□ JayneCatania
□ JulianaTilman
■ LaviniaHupp

DistrictAttorney
□ VickiNugent
□ ShanellDepalma
□ YvoneMoench

CountyTreasurer
□ AlphaGillespie
□ MorrisLoadholt
■ VondaFill

Sheriff
□ LoretteFeaster
□ VerniaHornstein
□ MelanyMillikan
□ AndriaHammes
■ MyrnaAlvarez
□ LatoyaLewallen

WHAT?

WHEN? WHAT?

COLUMN

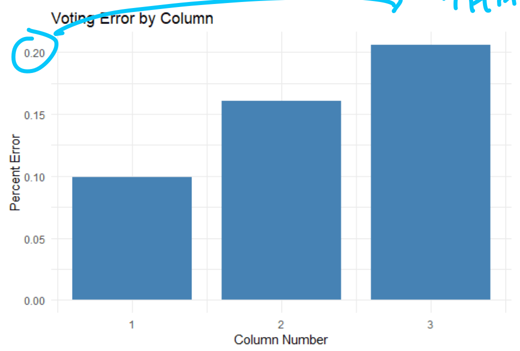
~~We now move on to analyzing the rate and nature of these errors.~~ The first statistic we are concerned about is the location of the model's errors across all ballots. For every race we have the exact y coordinate and column of each race, so we generated a heatmap of errors across columns and y position, where each bin collates all the races that occurred in a column within bins of 10 vertical pixels. Here the y position of a race is its header. This decision is why there is a blank area under the area at the top, as races have a minimum height implied by their minimum length of one candidate, so no races can ever be a certain distance from the top. These worries of impossible areas for races to appear in disappear and lead to a relatively uniform distribution of y position as y increases past this point because of the varied spacing and offsets that went into the simulation. For a measure of scale and significance, each box on this heatmap averages more than 500 races.

HUH? CONFUSING

WHAT DECISION?

One thing we may quickly observe is that there is a general trend of increasing error across columns, ~~and a quick bar plot~~ does indeed show this is true. In other words, races that are in later columns are more likely to be skipped

THAT'S A PROPORTION, NOT A %



DEFINE "% ERROR"
% OF WHAT?

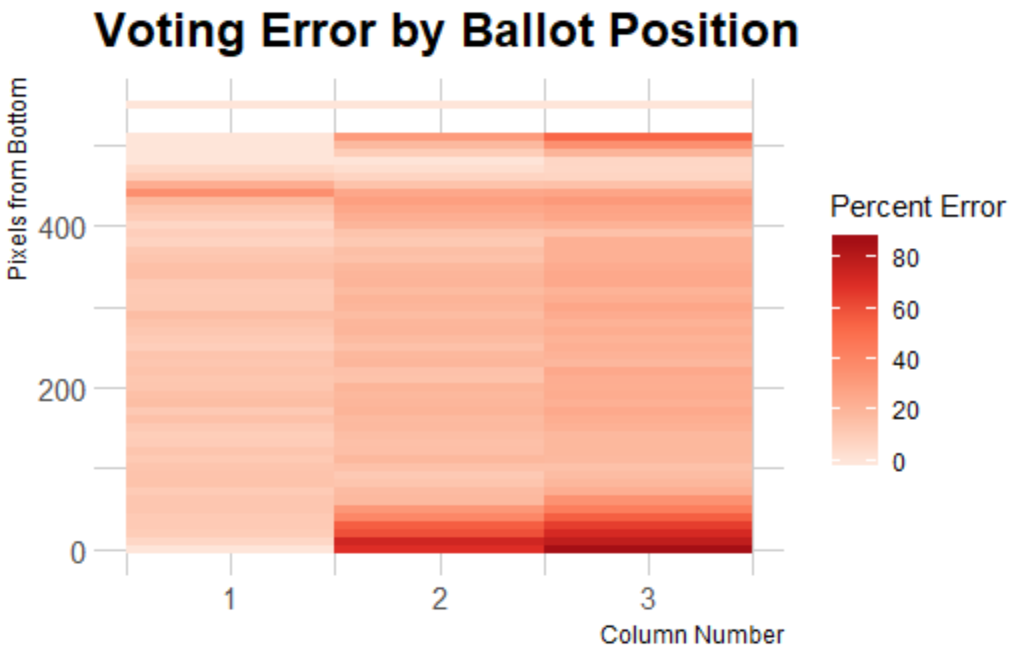
.

NOT AN OUTLIER. JUST EXPLAIN WHY

AWK

AWK

Of more interest are specific areas where errors are more likely. Notably, the very bottom right corner is frequently missed. This outlier does not shift the averages that much, as because of the nature of generating the ballot races are rarely placed there because usually they do not fit. However, if a short race is located there it is very likely that a person following this strategy will get to the last race on the left side and fall short of reaching the bottom.



CONFUSING

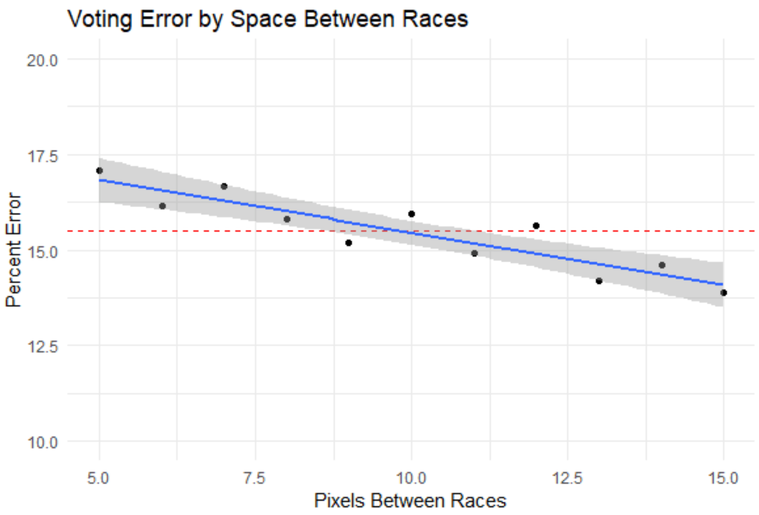
WHICH DIAGRAM

Finally, the last place to note on this diagram that has significant errors is directly under the race headers at the top. These occur because there is only a race there when there is a very short first race, and this opens the possibility that there will be a long race at the beginning and that second short race will be skipped. In other words, less conditions need to be met for a race at this position to be skipped. However, the situations that cause a skip here can cause a skip anywhere. In other words, our heatmap is also representing the fact that our model might run on a single ballot and identify problem spots anywhere, but these spots are getting lost in the noise. Thus, we move on to more closely analyzing what exactly causes skips.

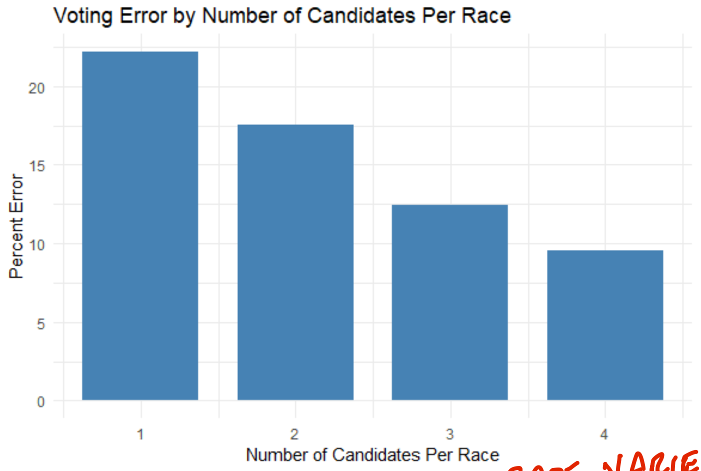
AMOUNT ?

EXACT #

We thus first examined the average error as we varied the number of vertical space between the end of each race and the beginning of the next. Again, using our dataset of 1000s of runs of the model, we found that as the space decreased, voting error increased (see figure x). This result suggests ballots that, for voters using a horizontal first voting strategy, the more cluttered a ballot is the more likely they are to miss a race.



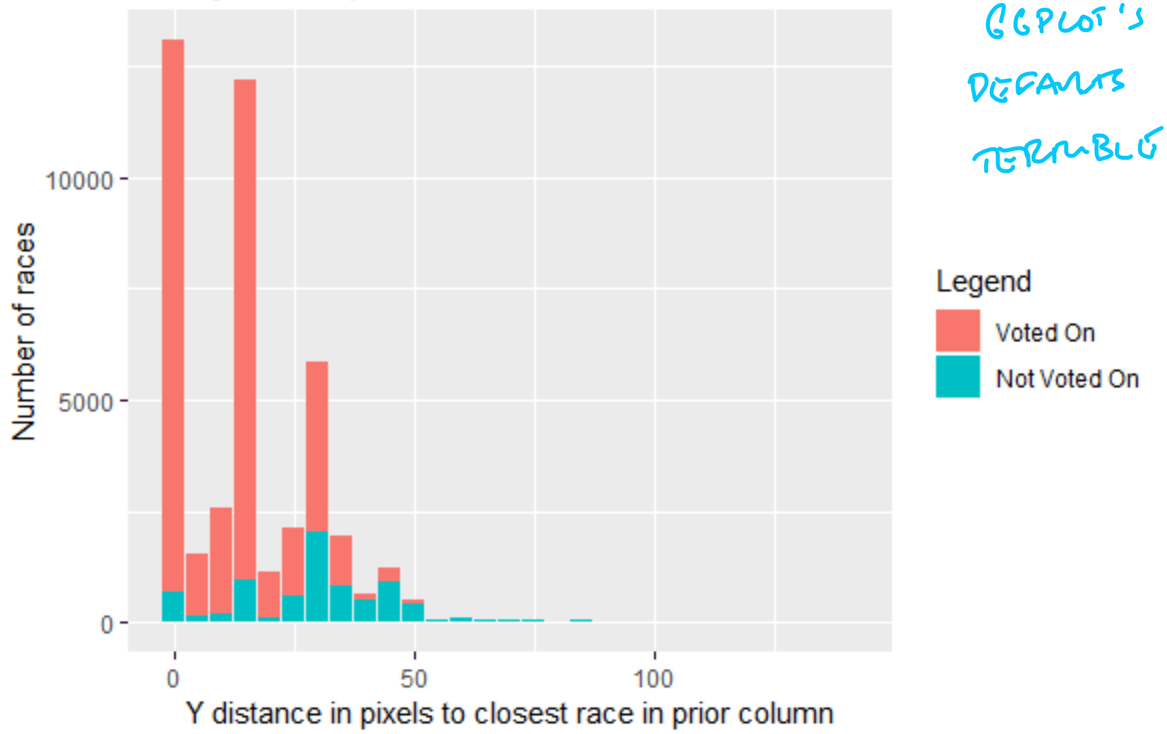
We also looked at how the length of a race was related to the chance it would be skipped and found similar results: as the length of a race decreased, its rate of being skipped increased (see figure x).



Finally, we looked at how ~~one more statistic varied with model error~~ ERROR RATE VARIED AS A FUNCTION OF: the vertical distance from a given race to the nearest race to it in the last column. Below, we plot bins of 5 pixels of this vertical distance versus both the number of races that were voted on in that bin and that were not. The stacked bar plot here is useful because it shows two things: that the chance a simulated voter missing a race increases as the closest distance to the last race increases, and that the number of races that are far from any prior race decreases as the distance increases. This graph more than any other illustrates the model's behavior and tendency to miss races that are not lined up in a row; actually building and running the simulation allows us to identify what these races are for any given ballot.

CAPTION, NOT TITLE

~~Voting Error by Closest Race Distance~~



GGPLOT'S DEFAULTS ARE TERRIBLE

We examined one more variable that led to no statistical difference: spacing within the race, between candidates and between the race header and the first candidate. This absence of a difference makes sense, as we built and used a 'perfect' visual grouping algorithm. Once the model found a race header, it could always find the candidates and the party.

A Github repository containing the documented model source code, videos of the model running, and the data from the simulate runs is located here: (make documented repository later).

SHOULD GO EARLY

Conclusions: COLON

~~Ultimately, these results fundamentally make sense~~. Races were more likely to be missed if they were smaller, out of line with the races in other columns, or more cramped overall. These are all characteristics of bad ballots. In other words, using a non-standard macronavigation strategy amplifies our ability to detect bad ballots. For instance, a strategy moving in the same direction as the races were originally placed might not mind if the races were very close together, but any other strategy would. Ballot designers need to cater to these less common strategies, so an ability to detect when ballots will cause systematic errors in voters using these strategies is crucial.

Indeed, we should note that the average error for this strategy is far higher than the average error for all voters, even assuming as we did a perfect micronavigation strategy. Most real voters probably use a more successful strategy. However, if even a subset of voters uses this strategy, or one like it, then we must account for them in our model, as a subset of voters can still have a deciding impact on a close race.

Thus, one of our next steps will be to entirely map the space of macronavigation strategies by running eye tracking experiments on human subjects voting on ballots. To implement these new strategies, we will need to expand the capabilities of ACT-R itself by extending the current visual grouping module to group objects in a hierarchy and by extending the options models have to visually navigate.

We also plan to build new strategies in the other modular categories of the model, including new ways for the model to encode the candidate, party, and race groups and new ways the model finds and clicks the circle corresponding to a candidate. Again, we will need to run studies to determine every variant behavior. Once we succeed, we will have a system that can dynamically build *any* voter from the voting strategy space. Each strategy will have a characteristic error pattern like we described in this paper, but more importantly can be run on novel ballots and determine if the error rate is above average.

Thus, while some of the findings may seem obvious, they must partly be viewed in the light of the wider project. Our model was able to vote on a wide array of ballots that looked hugely different and successfully make consistent errors. More than just characterizing the type of ballots and races that are more disposed to be skipped by a specific voter, these findings confirm the feasibility of attempting to eventually predict errors in novel ballots.