“Beating” Fitts’ Law

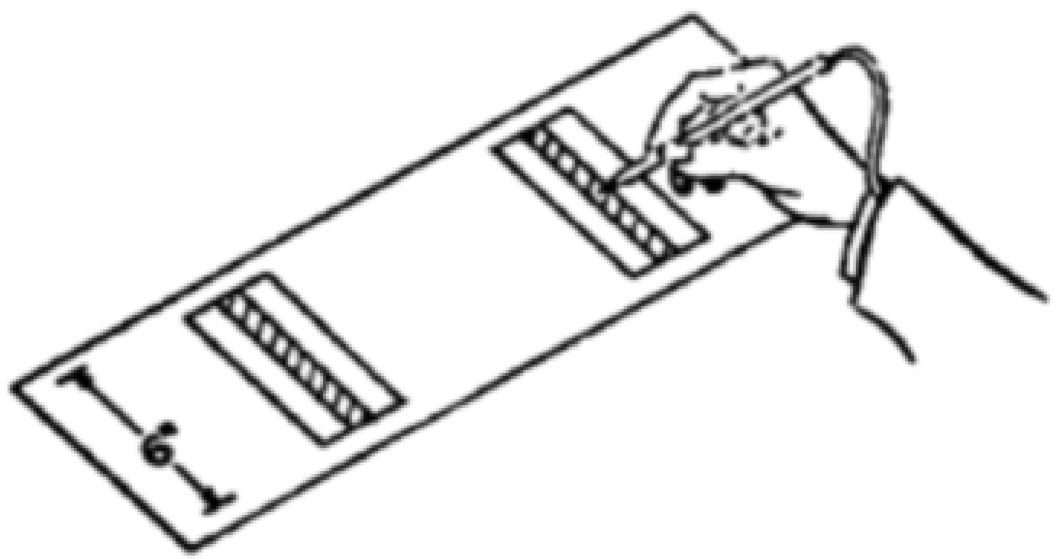
Jaime Ruiz

CS 349

July 22, 2009

Fitts' Law

- Published by Paul Fitts in 1954
- Most robust and highly adopted models of human movement.



Fitts' Law

$$MT = a + b\log_2\left(\frac{A}{W} + 1\right)$$

- *a and b are empirically defined constants*
- *A = Amplitude of movement (distance between start point and center of target)*
- *W = Constraining size of target*

Fitts' Law

$$MT = a + b \log_2 \underbrace{\left(\frac{A}{W} + 1 \right)}_{\text{Also Known as Index of Difficulty (ID)}}$$

- *a and b are empiri… s*
- *A = Amplitude of m… between start poi…)*
- *W = Constraining size of target*

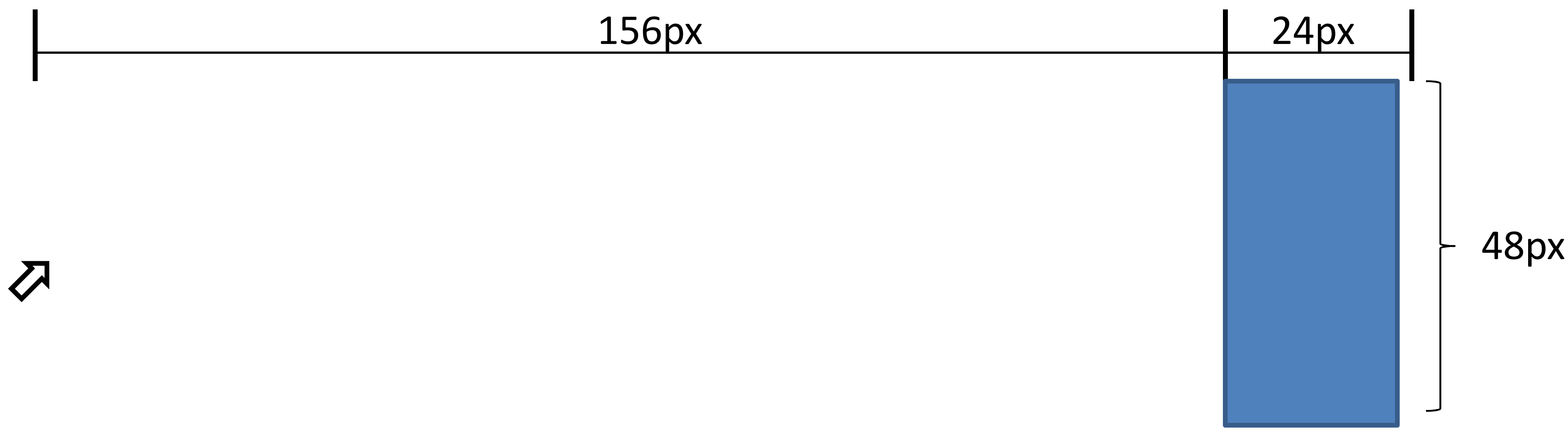
Figure 11. **Survey of Fitts' law performance characteristics from six studies on user input devices.**

Device	Study	Regression Coefficient[a] Intercept, *a* (ms)	Slope, *b* (ms/bit)	*IP*(bits/s)	*r*	Errors (%)	Comments
Eye tracker[b]	Ware & Mikaelian (1987)	680	73	13.7	—	8.5	Hardware button
Foot pedal	Drury (1975)	187	85	11.8	.97	<3.3	Experiment 2
Hand[c]	Fitts (1954)	12.8	94.7	10.6	.98	1.8	Tapping, 1-oz stylus
Mouse	Card, English, & Burr (1978)	1030	96	10.4	.91	5	
Eye tracker[b]	Ware & Mikaelian (1987)	790	97	10.3	—	22	On-screen button
Eye tracker[b]	Ware & Mikaelian (1987)	680	107	9.3	—	12	Dwell time
Helmet sight	Jagacinski & Monk (1985)	−268	199	5.0	.99	0	
Joystick	Jagacinski & Monk (1985)	−303	199	5.0	.99	0	Isometric; position contro
Joystick	Card, English, & Burr (1978)	990	220	4.5	.94	12	Isometric; velocity control
Joystick	Kantowitz & Elvers (1988)	−328	297	3.4	.62	25	Isometric, position, high
Joystick	Kantowitz & Elvers (1988)	−447	297	3.4	.76	25	Isometric, position, low g
Trackball	Epps (1986)	282	347	2.9	.93	0	
Mouse	Epps (1986)	108	392	2.6	.83	0	
Touchpad	Epps (1986)	181	434	2.3	.74	0	Absolute positioning
Joystick	Kantowitz & Elvers (1988)	−846	449	2.2	.84	25	Isometric, velocity, high
Joystick	Kantowitz & Elvers (1988)	−880	449	2.2	.85	25	Isometric, velocity, low g
Touchpad	Epps (1986)	−194	609	1.6	.70	0	Relative positioning
Joystick	Epps (1986)	−587	861	1.2	.81	0	Isometric; velocity control
Joystick	Epps (1986)	−560	919	1.1	.86	0	Displacement; velocity co

[a]$MT = a + b\,ID$; $IP = 1/b$. [b]Data inferred from plot. [c]Provided for comparison purposes only.

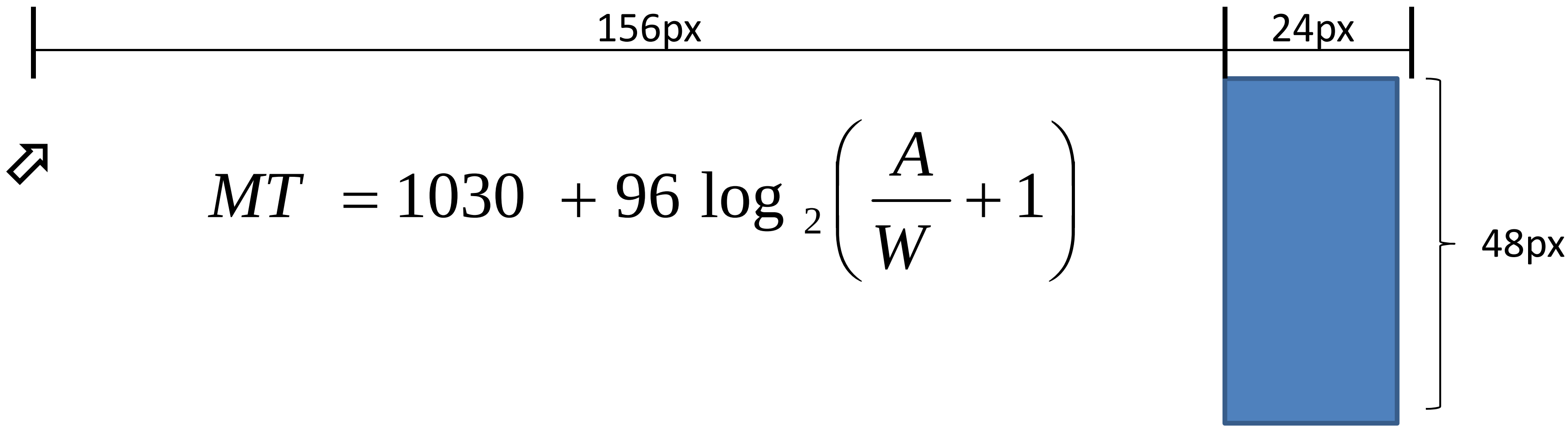
Examples

- Calculate movement time given a=1030, b=96



Examples

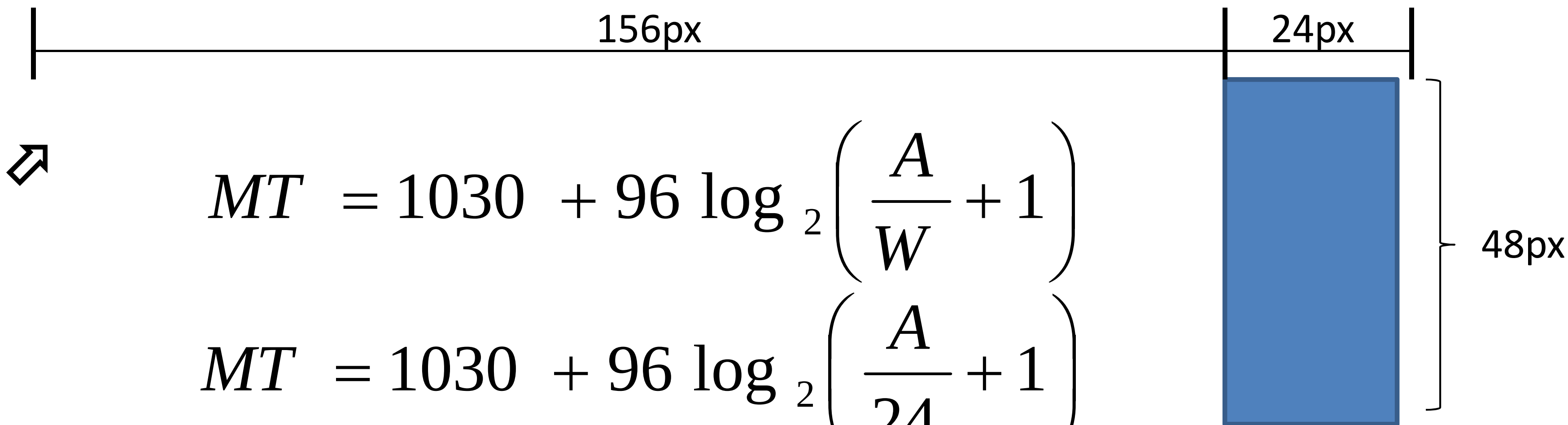
- Calculate movement time given a=1030, b=96



$$MT = 1030 + 96 \log_2\left(\frac{A}{W} + 1\right)$$

Examples

- Calculate movement time given a=1030, b=96

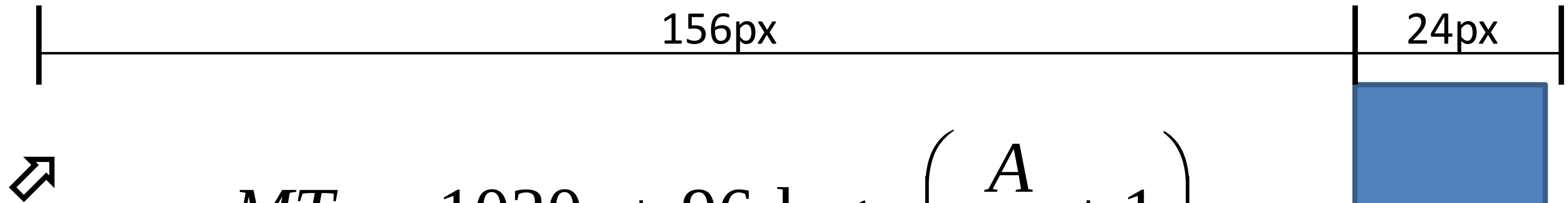


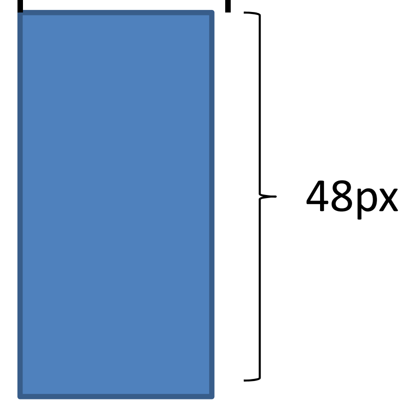
$$MT = 1030 + 96 \log_2\left(\frac{A}{W} + 1\right)$$

$$MT = 1030 + 96 \log_2\left(\frac{A}{24} + 1\right)$$

Examples

- Calculate movement time given a=1030, b=96





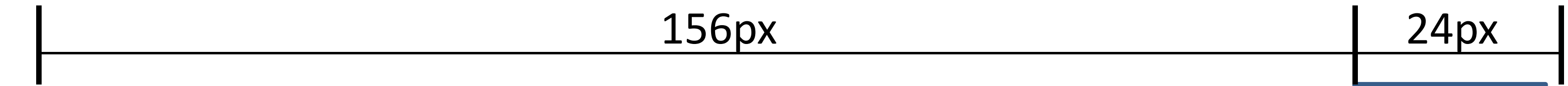
$$MT = 1030 + 96 \log_2\left(\frac{A}{W} + 1\right)$$

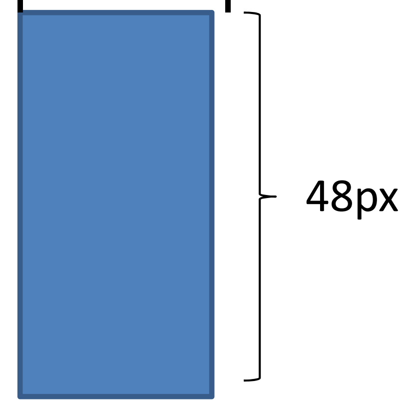
$$MT = 1030 + 96 \log_2\left(\frac{A}{24} + 1\right)$$

$$MT = 1030 + 96 \log_2\left(\frac{168}{24} + 1\right)$$

Examples

- Calculate movement time given a=1030, b=96





$$MT = 1030 + 96 \log_2\left(\frac{A}{W} + 1\right)$$

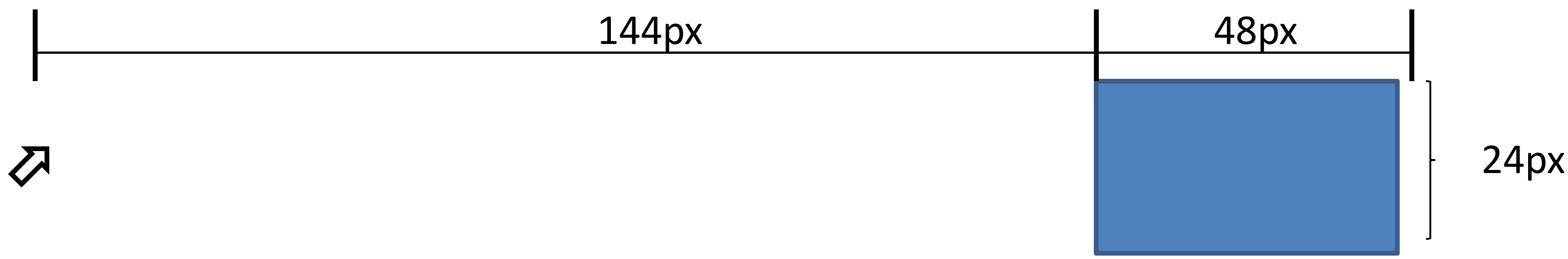
$$MT = 1030 + 96 \log_2\left(\frac{A}{24} + 1\right)$$

$$MT = 1030 + 96 \log_2\left(\frac{168}{24} + 1\right)$$

$$MT = 1030 + 96 \log_2(8) = 1318\ ms$$

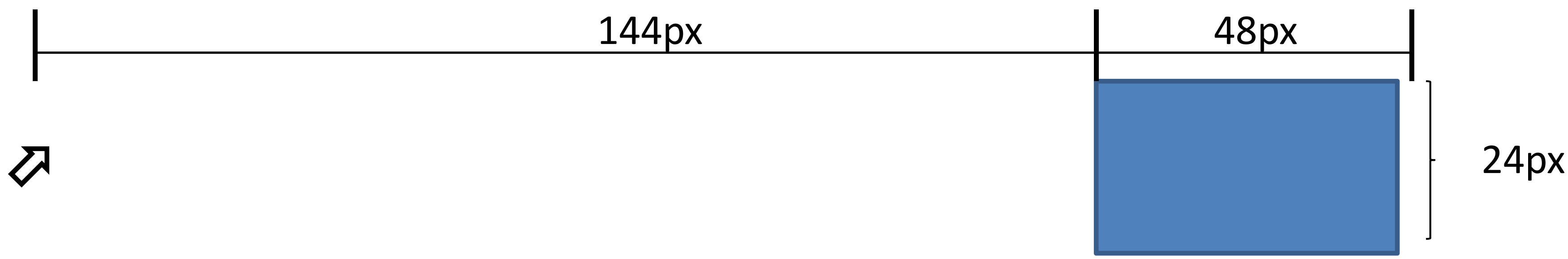
Examples

- Calculate movement time given a=1030, b=96



Examples

- Calculate movement time given a=1030, b=96



$$MT = 1030 + 96 \log_2\left(\frac{168}{24} + 1\right)$$

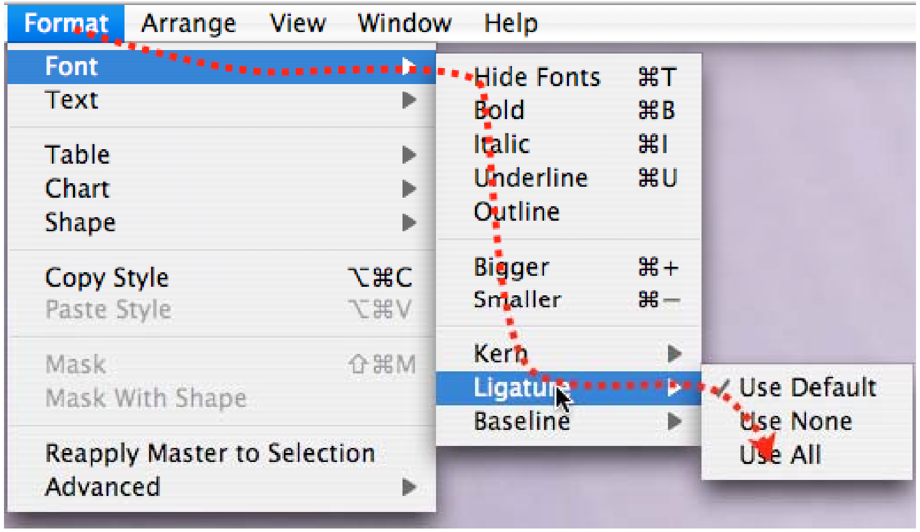
$$MT = 1030 + 96 \log_2(8) = 1318\ ms$$

Adaptations of Fitts' Law

- Steering Law

$$MT = a + b\left(\frac{A}{W}\right)$$

- W = height/width of tunnel
- A = amplitude/distance



Is it possible to “Beat” Fitts’ Law?

$$MT = a + b\log_2\left(\frac{A}{W} + 1\right)$$

Is it possible to "Beat" Fitts' Law?

$$MT = a + b\log_2\left(\frac{A}{W} + 1\right)$$

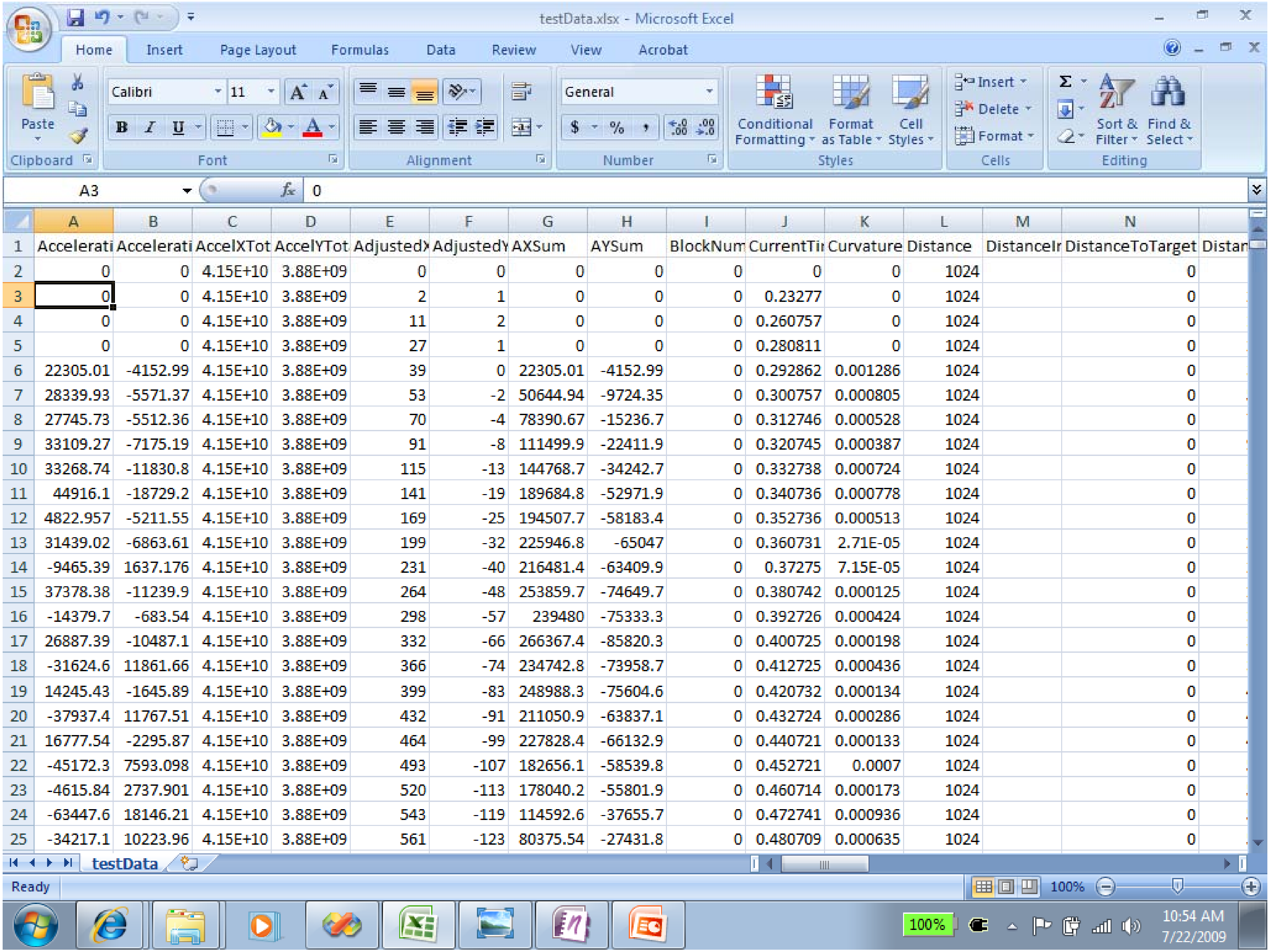
- Decrease distance to target
- Increase the width of target
- Or both

Drag & Pop (Baudisch et al. 2003)

- Minimize distance by bringing target closer

Object Pointing (Guiard et al 2004)

- Have mouse skip empty space

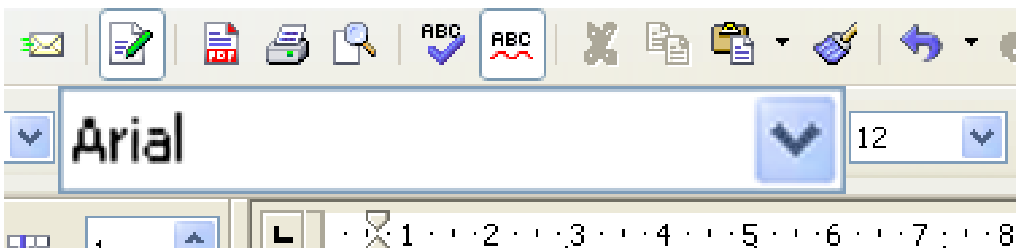


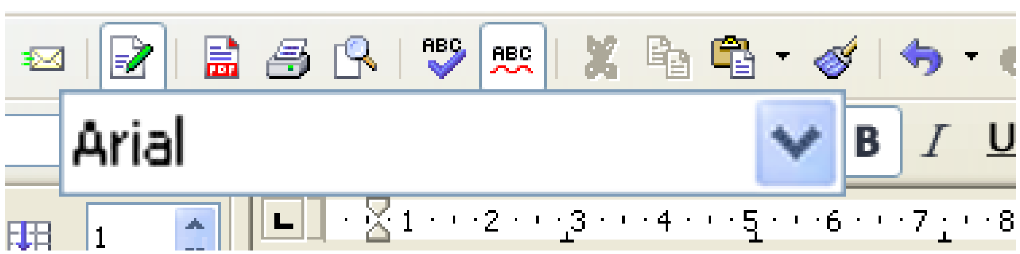
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	
1	Accelerati	Accelerati	AccelXTot	AccelYTot	AdjustedX	AdjustedY	AXSum	AYSum	BlockNum	CurrentTi	Curvature	Distance	DistanceIr	DistanceToTarget	Distan
2	0	0	4.15E+10	3.88E+09	0	0	0	0	0	0	0	1024		0	
3	0	0	4.15E+10	3.88E+09	2	1	0	0	0	0.23277	0	1024		0	
4	0	0	4.15E+10	3.88E+09	11	2	0	0	0	0.260757	0	1024		0	
5	0	0	4.15E+10	3.88E+09	27	1	0	0	0	0.280811	0	1024		0	
6	22305.01	-4152.99	4.15E+10	3.88E+09	39	0	22305.01	-4152.99	0	0.292862	0.001286	1024		0	
7	28339.93	-5571.37	4.15E+10	3.88E+09	53	-2	50644.94	-9724.35	0	0.300757	0.000805	1024		0	
8	27745.73	-5512.36	4.15E+10	3.88E+09	70	-4	78390.67	-15236.7	0	0.312746	0.000528	1024		0	
9	33109.27	-7175.19	4.15E+10	3.88E+09	91	-8	111499.9	-22411.9	0	0.320745	0.000387	1024		0	
10	33268.74	-11830.8	4.15E+10	3.88E+09	115	-13	144768.7	-34242.7	0	0.332738	0.000724	1024		0	
11	44916.1	-18729.2	4.15E+10	3.88E+09	141	-19	189684.8	-52971.9	0	0.340736	0.000778	1024		0	
12	4822.957	-5211.55	4.15E+10	3.88E+09	169	-25	194507.7	-58183.4	0	0.352736	0.000513	1024		0	
13	31439.02	-6863.61	4.15E+10	3.88E+09	199	-32	225946.8	-65047	0	0.360731	2.71E-05	1024		0	
14	-9465.39	1637.176	4.15E+10	3.88E+09	231	-40	216481.4	-63409.9	0	0.37275	7.15E-05	1024		0	
15	37378.38	-11239.9	4.15E+10	3.88E+09	264	-48	253859.7	-74649.7	0	0.380742	0.000125	1024		0	
16	-14379.7	-683.54	4.15E+10	3.88E+09	298	-57	239480	-75333.3	0	0.392726	0.000424	1024		0	
17	26887.39	-10487.1	4.15E+10	3.88E+09	332	-66	266367.4	-85820.3	0	0.400725	0.000198	1024		0	
18	-31624.6	11861.66	4.15E+10	3.88E+09	366	-74	234742.8	-73958.7	0	0.412725	0.000436	1024		0	
19	14245.43	-1645.89	4.15E+10	3.88E+09	399	-83	248988.3	-75604.6	0	0.420732	0.000134	1024		0	
20	-37937.4	11767.51	4.15E+10	3.88E+09	432	-91	211050.9	-63837.1	0	0.432724	0.000286	1024		0	
21	16777.54	-2295.87	4.15E+10	3.88E+09	464	-99	227828.4	-66132.9	0	0.440721	0.000133	1024		0	
22	-45172.3	7593.098	4.15E+10	3.88E+09	493	-107	182656.1	-58539.8	0	0.452721	0.0007	1024		0	
23	-4615.84	2737.901	4.15E+10	3.88E+09	520	-113	178040.2	-55801.9	0	0.460714	0.000173	1024		0	
24	-63447.6	18146.21	4.15E+10	3.88E+09	543	-119	114592.6	-37655.7	0	0.472741	0.000936	1024		0	
25	-34217.1	10223.96	4.15E+10	3.88E+09	561	-123	80375.54	-27431.8	0	0.480709	0.000635	1024		0	

Bubble Cursors (Grossman and Balakrishnan 2005)

- Video

Expanding Widgets





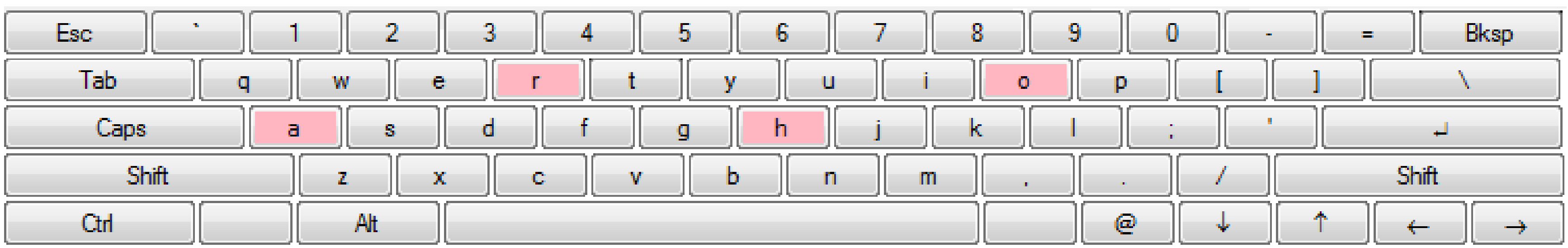
Predicting Endpoint

- Use laws of motion to try to predict endpoint
  - i.e. Minimum Jerk Law
- Initial algorithm
  - 40% correct
  - 40% +/- 1 target
  - 20% way off

Expanding Predictive Endpoint Cued Tablet Keyboard (EXPECT-K)

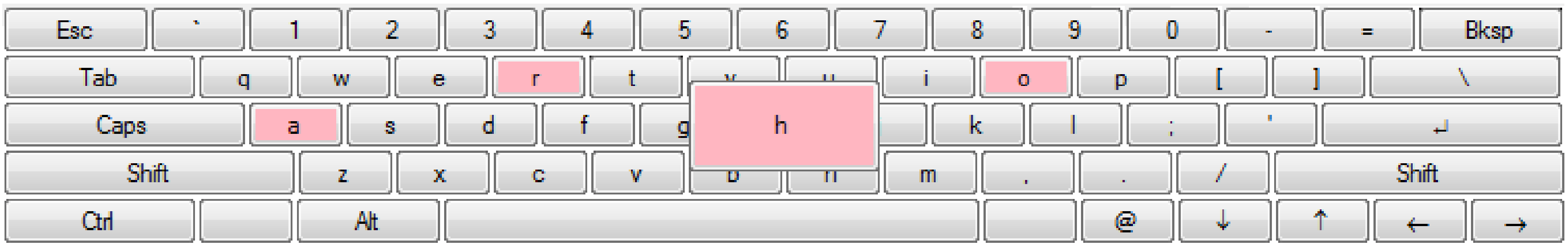
- The first virtual keyboard to incorporate endpoint prediction, target expansion and visual cues to speed text entry on Tablet PCs.

Visual Cues



- Keys highlighted according to tetra-gram model representing adjacent letter frequencies.
- The four keys representing the most frequent tetra-grams are highlighted
- Tetra-gram model is updated continuously allowing the model to adjust to the individual’s language usage.

Expanding Keys



- Expansion of the user's intended key is made possible by a real-time implementation of the Lank et al. endpoint prediction algorithm.
- The result from the endpoint predictor, in conjunction with the tetra-gram letter frequencies, is used to predict which key should be expanded.

VIDEO

Participate

- 30-60 minutes
- Some pay $10
- Need lots of people between now and September.
- If interested sign the sheet being passed around or send an email to jgruiz@uwaterloo.ca