

1 **Demonstrating Headphone-Sensed, Accessibility-Informed Head Pointing with**
2 **Snapping for Inclusive Interaction**

3 **MYKHAILO TARASENKO**, National University of Kyiv-Mohyla Academy, Ukraine and MacPaw, Ukraine

4 **IRYNA PASTUKHOVA**, MacPaw, Ukraine

5 **OLEKSANDR FRANKIV**, National University of Kyiv-Mohyla Academy, Ukraine and MacPaw, Ukraine

6 **ANASTASIIA SATARENKO**, MacPaw, Ukraine

7 **NATALIA STULOVA**, MacPaw, Ukraine

8 **SERGII KRYVOBLOTSKYI**, MacPaw, Ukraine

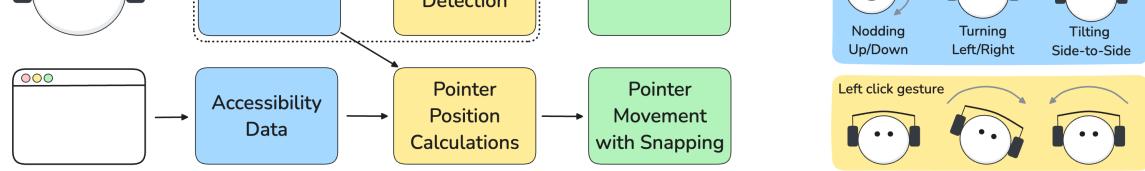
9

10

11

12

13



14
15 Fig. 1. General head pointing approach architecture with detailisation of captured data and detected gesture visualization
16
17
18

19 The mouse pointer is central to direct-manipulation graphical user interfaces, and modern desktop operating systems provide
20 accessibility features that enable pointer control via eye or head movement. Existing solutions, such as Eye Control on Windows and
21 Head Pointer on macOS, rely on continuous video capture, which makes them sensitive to lighting conditions and user position and
22 raises privacy concerns. In this work, we demonstrate a head-pointing approach based on head movements captured by the gyroscope
23 and accelerometer sensors of commercial headphones. To improve pointing precision, we implement pointer snapping that leverages
24 accessibility information from application user interfaces. This approach is independent of camera placement and lighting conditions,
25 offers privacy advantages, and requires no specialized hardware beyond commonly used headphones, supporting more inclusive and
26 accessible interaction. Our demo highlights how reusing existing accessibility infrastructure can support more inclusive pointing
27 interactions and contribute to creating more accessible interactive systems.

28
29
30
31
32
33
34
35
36 CCS Concepts: • **Human-centered computing** → **Pointing**; **Gestural input**; **Pointing devices**; **Accessibility systems and tools**;
37 **Systems and tools for interaction design**.

38 Additional Key Words and Phrases: Head-based pointing, Head-based gestures, Accessibility, Gyroscope, Accelerometer, macOS

39
40
41 Authors' addresses: **Mykhailo Tarasenko**, National University of Kyiv-Mohyla Academy, Kyiv, Ukraine and MacPaw, Kyiv, Ukraine, ms.tarasenko@
42 ukma.edu.ua; **Iryna Pastukhova**, MacPaw, Kyiv, Ukraine, iryna.p@macpaw.com; **Oleksandr Frankiv**, National University of Kyiv-Mohyla Academy, Kyiv,
43 Ukraine and MacPaw, Kyiv, Ukraine, o.frankiv@ukma.edu.ua; **Anastasiia Satarenko**, MacPaw, Kyiv, Ukraine, an.sat@macpaw.com; **Natalia Stulova**,
44 MacPaw, Kyiv, Ukraine, nata.stulova@macpaw.com; **Sergii Kryvoblotskyi**, MacPaw, Kyiv, Ukraine, krivoblotsky@macpaw.com.

45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
55310
55311
55312
55313
55314
55315
55316
55317
55318
55319
55320
55321
55322
55323
55324
55325
55326
55327
55328
55329
55330
55331
55332
55333
55334
55335
55336
55337
55338
55339
55340
55341
55342
55343
55344
55345
55346
55347
55348
55349
55350
55351
55352
55353
55354
55355
55356
55357
55358
55359
55360
55361
55362
55363
55364
55365
55366
55367
55368
55369
55370
55371
55372
55373
55374
55375
55376
55377
55378
55379
55380
55381
55382
55383
55384
55385
55386
55387
55388
55389
55390
55391
55392
55393
55394
55395
55396
55397
55398
55399
553100
553101
553102
553103
553104
553105
553106
553107
553108
553109
553110
553111
553112
553113
553114
553115
553116
553117
553118
553119
553120
553121
553122
553123
553124
553125
553126
553127
553128
553129
553130
553131
553132
553133
553134
553135
553136
553137
553138
553139
553140
553141
553142
553143
553144
553145
553146
553147
553148
553149
553150
553151
553152
553153
553154
553155
553156
553157
553158
553159
553160
553161
553162
553163
553164
553165
553166
553167
553168
553169
553170
553171
553172
553173
553174
553175
553176
553177
553178
553179
553180
553181
553182
553183
553184
553185
553186
553187
553188
553189
553190
553191
553192
553193
553194
553195
553196
553197
553198
553199
553200
553201
553202
553203
553204
553205
553206
553207
553208
553209
553210
553211
553212
553213
553214
553215
553216
553217
553218
553219
553220
553221
553222
553223
553224
553225
553226
553227
553228
553229
553230
553231
553232
553233
553234
553235
553236
553237
553238
553239
553240
553241
553242
553243
553244
553245
553246
553247
553248
553249
553250
553251
553252
553253
553254
553255
553256
553257
553258
553259
553260
553261
553262
553263
553264
553265
553266
553267
553268
553269
553270
553271
553272
553273
553274
553275
553276
553277
553278
553279
553280
553281
553282
553283
553284
553285
553286
553287
553288
553289
553290
553291
553292
553293
553294
553295
553296
553297
553298
553299
553300
553301
553302
553303
553304
553305
553306
553307
553308
553309
553310
553311
553312
553313
553314
553315
553316
553317
553318
553319
553320
553321
553322
553323
553324
553325
553326
553327
553328
553329
553330
553331
553332
553333
553334
553335
553336
553337
553338
553339
5533310
5533311
5533312
5533313
5533314
5533315
5533316
5533317
5533318
5533319
55333110
55333111
55333112
55333113
55333114
55333115
55333116
55333117
55333118
55333119
553331110
553331111
553331112
553331113
553331114
553331115
553331116
553331117
553331118
553331119
5533311110
5533311111
5533311112
5533311113
5533311114
5533311115
5533311116
5533311117
5533311118
5533311119
55333111110
55333111111
55333111112
55333111113
55333111114
55333111115
55333111116
55333111117
55333111118
55333111119
553331111110
553331111111
553331111112
553331111113
553331111114
553331111115
553331111116
553331111117
553331111118
553331111119
5533311111110
5533311111111
5533311111112
5533311111113
5533311111114
5533311111115
5533311111116
5533311111117
5533311111118
5533311111119
55333111111110
55333111111111
55333111111112
55333111111113
55333111111114
55333111111115
55333111111116
55333111111117
55333111111118
55333111111119
553331111111110
553331111111111
553331111111112
553331111111113
553331111111114
553331111111115
553331111111116
553331111111117
553331111111118
553331111111119
5533311111111110
5533311111111111
5533311111111112
5533311111111113
5533311111111114
5533311111111115
5533311111111116
5533311111111117
5533311111111118
5533311111111119
55333111111111110
55333111111111111
55333111111111112
55333111111111113
55333111111111114
55333111111111115
55333111111111116
55333111111111117
55333111111111118
55333111111111119
553331111111111110
553331111111111111
553331111111111112
553331111111111113
553331111111111114
553331111111111115
553331111111111116
553331111111111117
553331111111111118
553331111111111119
5533311111111111110
5533311111111111111
5533311111111111112
5533311111111111113
5533311111111111114
5533311111111111115
5533311111111111116
5533311111111111117
5533311111111111118
5533311111111111119
55333111111111111110
55333111111111111111
55333111111111111112
55333111111111111113
55333111111111111114
55333111111111111115
55333111111111111116
55333111111111111117
55333111111111111118
55333111111111111119
553331111111111111110
553331111111111111111
553331111111111111112
553331111111111111113
553331111111111111114
553331111111111111115
553331111111111111116
553331111111111111117
553331111111111111118
553331111111111111119
5533311111111111111110
5533311111111111111111
5533311111111111111112
5533311111111111111113
5533311111111111111114
5533311111111111111115
5533311111111111111116
5533311111111111111117
5533311111111111111118
5533311111111111111119
55333111111111111111110
55333111111111111111111
55333111111111111111112
55333111111111111111113
55333111111111111111114
55333111111111111111115
55333111111111111111116
55333111111111111111117
55333111111111111111118
55333111111111111111119
553331111111111111111110
553331111111111111111111
553331111111111111111112
553331111111111111111113
553331111111111111111114
553331111111111111111115
553331111111111111111116
553331111111111111111117
553331111111111111111118
553331111111111111111119
5533311111111111111111110
5533311111111111111111111
5533311111111111111111112
5533311111111111111111113
5533311111111111111111114
5533311111111111111111115
5533311111111111111111116
5533311111111111111111117
5533311111111111111111118
5533311111111111111111119
55333111111111111111111110
55333111111111111111111111
55333111111111111111111112
55333111111111111111111113
55333111111111111111111114
55333111111111111111111115
55333111111111111111111116
55333111111111111111111117
55333111111111111111111118
55333111111111111111111119
553331111111111111111111110
553331111111111111111111111
553331111111111111111111112
553331111111111111111111113
553331111111111111111111114
553331111111111111111111115
553331111111111111111111116
553331111111111111111111117
553331111111111111111111118
553331111111111111111111119
5533311111111111111111111110
5533311111111111111111111111
5533311111111111111111111112
5533311111111111111111111113
5533311111111111111111111114
5533311111111111111111111115
5533311111111111111111111116
5533311111111111111111111117
5533311111111111111111111118
5533311111111111111111111119
55333111111111111111111111110
55333111111111111111111111111
55333111111111111111111111112
55333111111111111111111111113
55333111111111111111111111114
55333111111111111111111111115
55333111111111111111111111116
55333111111111111111111111117
55333111111111111111111111118
5

53 **ACM Reference Format:**

54 Mykhailo Tarasenko, Iryna Pastukhova, Oleksandr Frankiv, Anastasiia Satarenko, Nataliia Stulova, and Sergii Kryvoblotskyi. 2026.
 55 Demonstrating Headphone-Sensed, Accessibility-Informed Head Pointing with Snapping for Inclusive Interaction. 1, 1 (January 2026),
 56 7 pages. <https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

57 **1 INTRODUCTION AND MOTIVATION**

58 Mouse pointer and point-and-click workflows still remain the default expected way to interact with direct manipulation
 59 graphical user interfaces on personal computers and laptops. For computer users with impaired mobility, or for hands-
 60 free interaction scenarios in general, desktop operating systems vary in accessibility features and peripheral devices
 61 supported to accept alternative pointer controls. In Windows, the desktop OS with the largest user base worldwide, an
 62 eye mouse is available via Eye control technology. It relies solely on tracking the eye movement with a camera, and for
 63 pointer actions, the user is presented with a movable launchpad bar with predefined actions selected by dwelling the
 64 eye gaze on them. Moreover, its use is limited [9] “in locations with a lot of sunlight. Additionally, eye tracking works
 65 differently depending on eye color, eye size, or eye shape.” On macOS, the second most used desktop OS, the Head
 66 Pointer technology embodies the face mouse approach. It also uses a camera, processing user face expressions and head
 67 movement for cursor actions and position change, respectively. Limitations [7] of this technology include “the ambient
 68 lighting being too bright or dark, the user being too close to or far from the camera, or them not being centered in
 69 front of the screen”. Moreover, reliance of both solutions on camera feed requires either a built-in camera, an additional
 70 camera, or a camera-equipped device, and introduces user privacy concerns that need to be mitigated additionally.
 71

72 Among the head-based pointing methods, head mouse approach is another possible implementation of a relative
 73 pointing device. Research to date shows that when head mobility is a possible scenario, relative head-based pointer
 74 controls are preferred over absolute eye gaze-based ones [1, 4, 5]. Current head pointing solutions differ in the hardware
 75 used to capture head movement, including head-based cameras [13], optical sensors on the neck [6], head and mouth
 76 pieces [3], or glass-mounted sensors [12]. Recently, modern headphones were considered as a general-purpose input
 77 device [10]. While the proposed solutions differ in their implementations of pointer actions, the use of gyroscope and
 78 accelerometer sensors has become a shared pattern, nowadays also present in commercial head- and earphones.

79 In this work, we demonstrate an approach to extend the capabilities of an existing wearable device, like headphones,
 80 to the task of head pointing. While only using two sensors for head position tracking, we take advantage of application
 81 user interface (UI) annotations, which are available in macOS as a part of its accessibility features, as an information
 82 source to improve pointing precision, naming it pointer snapping and making it informed, aligning with the user
 83 interaction intent. Using accessibility information instead of gaze allows for avoiding limitations of the environment
 84 and focusing on actual interaction. We implement left and right mouse click actions with head gestures and showcase
 85 user learning and app interaction.

94 **2 BACKGROUND**

95 We briefly outline the relevant for our approach software and hardware components of the Apple ecosystem.

96 Accessibility-related functionality in computer system interfaces helps people with different types of needs, such
 97 as vision, speech, mobility, cognitive, and hearing, to interact with their devices. Accessibility¹ framework in macOS
 98 supports developers in building systems that address a wide range of access needs through a comprehensive set of APIs.
 99 One of the framework features is the accessibility tree, like one shown in Figure 2 (right), that represents an app’s user
 100

101 ¹<https://developer.apple.com/documentation/accessibility>

102 Manuscript submitted to ACM



Fig. 2. A match between UI elements in a Calculator app and the respective JSON of the accessibility tree (the “=” button fragment)

interface and is used by assistive technologies like VoiceOver² to parse the UI hierarchy of the currently focused window opened on the user’s desktop.

The other core framework of interest to us is CoreMotion³ that processes motion- and environment-related data from the onboard hardware of wearable devices. Our approach supports all headphones that are compatible with the CoreMotion’s CMHeadphoneMotionManager class according to the isDeviceMotionAvailable property. This class provides accelerometer, gyroscope, and other position data via a specific API. All headphones that support Apple’s Spatial Audio technology with dynamic head tracking on the device are compatible with this class⁴.

3 IMPLEMENTATION

Receiving data from headphones. To implement pointer movement using headphones that support Spatial Audio, we develop an application that uses the CoreMotion framework. Using its class CMHeadphoneMotionManager, we start tracking motion updates of the headphones by calling a method .startDeviceMotionUpdates(). As a result, our application starts receiving information about changes in the headphones’ position in space. This information is encapsulated in a CMDeviceMotion class, which gives access to different data, such as attitude, rotationRate, userAcceleration etc. Here, attitude is a struct which represents the pitch, yaw, and roll aspects of motion, as shown in Figure 1.

Calibration and coordinate mapping. For our approach, headphone motion data must be mapped to the screen coordinates. In the considered setup, the yaw value corresponds to the horizontal coordinate (x), and the pitch to the vertical coordinate (y). Thus, for any pointer position (x, y) , we have $x \in [0, width]$ and $y \in [0, height]$, where $width$ and $height$ are the screen dimensions. We assume that, on average, people can rotate their heads within a range of $\pm 30^\circ$ (approximately ± 0.5 rad). Applying the linear interpolation, for the given yaw value, the x coordinate can be computed as $x = x_{min} + t \cdot (x_{max} - x_{min})$ with $t = (yaw - yaw_{min}) / (yaw_{max} - yaw_{min}) = (yaw - (-0.5)) / (0.5 - (-0.5)) = yaw + 0.5$, and consequently, $x = 0 + (yaw + 0.5) \cdot width = (yaw + 0.5) \cdot width$. By analogy, for any pitch value, the y coordinate $y = y_{min} + (pitch + 0.5) \cdot height = (pitch + 0.5) \cdot height$.

To generalize our approach and reduce dependence on the user distance from the screen, the yaw and pitch values are calibrated before applying interpolation. For this, the user is asked to approach four near-corner points on the screen

²<https://developer.apple.com/documentation/accessibility/voicover>

³<https://developer.apple.com/documentation/coremotion>

⁴<https://developer.apple.com/videos/play/wwdc2023/10179/>



Fig. 3. Calibration process starts in a window

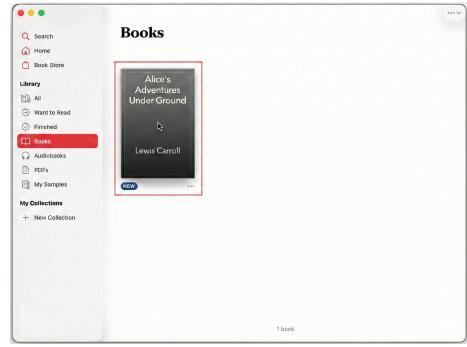


Fig. 4. Books app with pointer snapping

(see Figure 3) with the head pointer, providing user-specific extreme yaw and pitch values. Using the corresponding spans allows us to adjust the observed yaw and pitch values to the agreed-upon “universal” range above, ± 0.5 rad.

Pointer snapping approach. Precise hovering, maintaining focus, and interacting with UI elements, especially small or closely spaced ones, can be challenging with head pointing. To address this issue, we introduce the snapping pointer approach. We equip each UI element with a targeting neighborhood that slows the pointer, snaps it, and targets it towards the center. Directing a pointer to the intended UI element requires determining the precise position and size of the closest available element. We continuously monitor the change of the active application to capture the user’s context at each point in time. We traverse the accessibility tree of the active application to obtain UI elements, along with their positions on the screen, sizes, roles, actions, and other attributes. The Accessibility API allows us to respond to UI changes immediately. We further filter interface elements based on their accessibility roles such as AXButton, AXPopUpButton, and supported actions such as kAXPressAction, kAXShowMenuAction, relevant to pointer interaction.

Another consideration is the pointer snap itself. To make focusing on the UI element easier, we increase the motion threshold in the targeting neighborhood and highlight the element as it is approached. If the pointer is directly snapped to and released from the element’s neighborhood, from the user’s perspective, it feels like jumping and resistance. Also, even though there is almost always only one nearest element (target) at any given moment, in the case of close elements, because the pointer is very sensitive, even a small, often unintentional move can change the target. These rapid changes of the target generate oscillation between the close or intersecting neighborhoods, something like a “ping-pong” effect. All the above effects are mitigated by introducing the hysteresis, such that the threshold for entering the element is lower than for exiting it. At the same time, if the pointer enters the element itself rather than only its corresponding targeting neighborhood, it is selected immediately, enabling smooth navigation through list-like interfaces.

Gestures implementation. The ability to move the pointer alone does not provide complete interaction with the UI. For the purpose of the approach demonstration, we implement left- and right-click actions as a minimal set of interactions. We select head tilts for this purpose, as they are intuitive and rely on the motion roll values that are not yet utilized by our approach, unlike pitch and yaw. Our calibration provides the maximum and minimum roll values, which represent the range of neutral head positions. We will refer to them as the stability interval. We define a click gesture as a time-bounded (0.8s) sharp tilt toward the corresponding side initiated inside the stability interval, followed

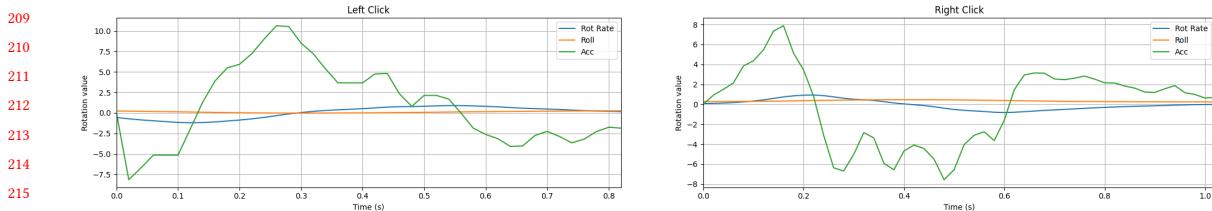


Fig. 5. Roll, rotation rate, and acceleration signals for left and right click gestures

by a rapid return to the starting position. Since the end point of the gesture does not always coincide precisely with the start point, half the stability interval tolerance is allowed.

Figure 5 presents the roll, rotation rate, and acceleration signals for the left and right clicks, respectively, highlighting that acceleration provides the most reliable indicator of the user’s gesture intent. We observe that during a left-click gesture, acceleration changes sign from negative to positive. In contrast, during a right-click gesture, the opposite pattern occurs, enabling a reliable distinction between them. Thus, a left click is modelled as a sequence of three states in terms of acceleration: (1) neutral (head roll is within the stability interval), (2) tilt toward the left side, initiated by passing the acceleration negative threshold, (3) return to neutral, detected by acceleration change to positive and neutral head position adjusted according to the tolerance, as shown in Figure 1 (right bottom). This sequence must occur within a predefined time window. By analogy, the right click is identified using the inverse acceleration pattern. The acceleration threshold was initially set to the mean of the minimum absolute values observed across 200 recorded left- and right-click events from one of the authors, and then tuned through empirical evaluation to improve generalization with the final value of 7.0 rad/s^2 .

Performing the gestures obviously induces significant changes in pitch and yaw, moving the pointer away from the intended element and resulting in clicks at unpredictable points. To prevent this, we further refine the snapping pointer with a freezing pointer effect. After holding an element in focus for some time, the pointer temporarily freezes, allowing the user to perform a gesture with greater precision. The focusing and freezing phases involve highlighting the corresponding element with different colors to provide visual feedback to the user (e.g., see Figure 4). In our experiments, a duration of 700 ms was sufficient to consider the element approach intentional without hindering natural movement, while 1.5 s of freezing allowed for deciding on and performing a gesture.

4 DEMO

To let users experience the proposed head pointing technique, we developed two training mini-games (Figure 6).

We’ve developed a mini-game called **Tiles** to introduce the user to basic pointer control and clicking gestures. This application allows users to capture an image with a built-in laptop camera and use it as a puzzle. The image is divided into a grid of 3×3 , and the tiles are randomly shuffled. Users select and place tiles using the pointer input. We adapted the interaction to a click-based mechanism: a tile attaches to the pointer on a left click and can be released with the same click while hovering over the target location.

We developed the **Banners** mini-game to familiarize users with the effective use of the snapping pointer. It simulates the appearance of multiple intrusive advertisement-like windows on the desktop. Users are instructed to close all banners by clicking on a system-like close button on the header of each window. The close buttons are intentionally

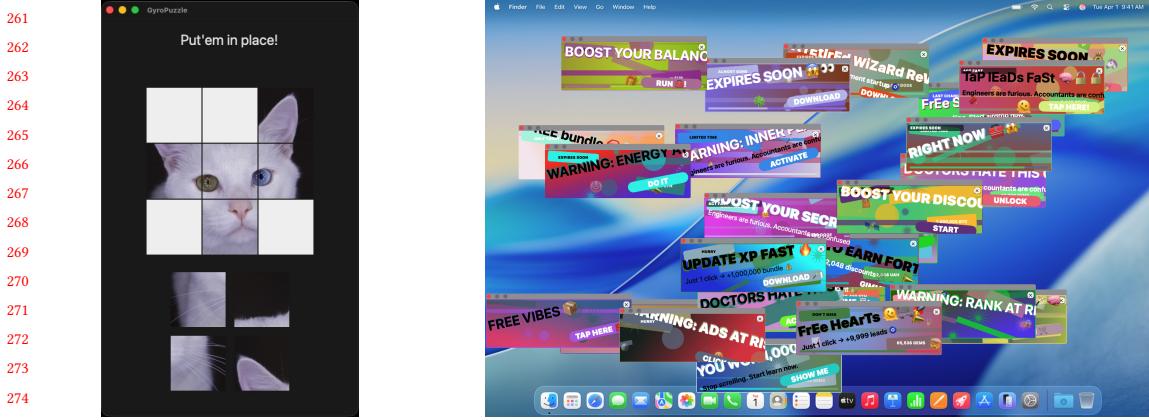


Fig. 6. **Tiles** (left) and **Banners** (right) mini-games' interfaces

small, comparable to the default size in macOS. When the pointer approaches a button, a snapping mechanism attracts the pointer toward the target, facilitating reliable selection.

5 DISCUSSION AND CONCLUSION

We demonstrated an approach to desktop head pointing for mouse pointer control that leverages accessibility annotations in direct-manipulation graphical user interfaces. While our implementation is presented in the context of the Apple ecosystem, the proprietary technologies are not foundational to it. Instead, we showcase how leveraging the developer-facing parts of OS frameworks and building on existing system support for sensor data and accessibility information allows us to complement and extend accessibility functionalities. Our work highlights how accessibility technologies can benefit a broad range of users beyond those requiring accommodations, and we hope it will encourage operating system and application developers to further adopt and maintain accessibility annotations and functionalities.

In the hardware part, our approach relies on headphones and earphones equipped with a combination of accelerometer and gyroscope sensors. These two sensors are an integral part of various headset-based solutions to head pointing [2, 8, 11], and while full headsets allow for richer interactions with the desktop computers and other electronic devices, we believe we have shown that even with minimal hardware use full point-and-click workflows become achievable. Another consideration we had in mind was the reuse of existing technologies and electronics. By extending the use of widely available consumer hardware, our approach also points toward more sustainable interaction design practices, which could minimize the e-waste problem.

ACKNOWLEDGMENTS

We thank the Armed Forces of Ukraine for providing security to complete this work. We thank Mariia Kak and Vladyslav Makariuk for their help with accompanying video production, and Kseniia Dobrieva and Anastasiia Kyiashko for consultations on legal matters. This is an independent research publication, and it has not been authorized, sponsored, or otherwise approved by Apple Inc. Apple, Apple Books, AirPods Pro, and AirPods Max are trademarks of Apple Inc.

Manuscript submitted to ACM

313 REFERENCES

314 [1] R. Bates and H. O. Istance. 2003. Why are eye mice unpopular? A detailed comparison of head and eye controlled assistive technology pointing
 315 devices. *Universal Access in the Information Society* 2, 3 (2003), 280–290. <https://doi.org/10.1007/s10209-003-0053-y>

316 [2] Aaron Castillo, Graciela Cortez, David Diaz, Rayton Espíritu, Krystle Ilisastigui, Bryce O'Bard, and Kiran George. 2016. Hands free mouse. In *2016
 317 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. 109–114. <https://doi.org/10.1109/BSN.2016.7516242>

318 [3] Rodrigo Duarte, Nuno Vieira Lopes, and Paulo Jorge Coelho. 2025. A Low-Cost Head-Controlled and Sip-and-Puff Mouse: System Design and
 319 Preliminary Findings. *Electronics* 14, 24 (2025). <https://doi.org/10.3390/electronics14244953>

320 [4] David G. Evans, R. Drew, and Paul Blenkhorn. 2000. Controlling mouse pointer position using an infrared head-operated joystick. *IEEE Transactions
 321 on Rehabilitation Engineering* 8, 1 (2000), 107–117. <https://doi.org/10.1109/86.830955>

322 [5] Gareth Evans and Paul Blenkhorn. 1999. A Head Operated Joystick—Experience with Use. (1999). Technical report. <https://eric.ed.gov/?id=ED430330>.

323 [6] Ali Heydarigori, S. M. Safavi, Ch. T. Lee, and P. H. Chou. 2017. Head-mouse: A simple cursor controller based on optical measurement of head tilt.
 324 In *2017 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*. 1–5. <https://doi.org/10.1109/SPMB.2017.8257058>

325 [7] Apple Inc. 2025. *Move the pointer using head pointer on Mac*. Online. <https://support.apple.com/guide/mac-help/use-head-pointer-mchlbd4782b/26/mac/26>.

326 [8] Mohammad Ridwan Kabir, Mohammad Ishrak Abedin, Rizvi Ahmed, Saad Bin Ashraf, Hasan Mahmud, and Md. Kamrul Hasan. 2024. Auxilio and
 327 Beyond: Comparative Evaluation, Usability, and Design Guidelines for Head Movement-based Assistive Mouse Controllers. arXiv:2210.04483 [cs.HC]
 328 <https://arxiv.org/abs/2210.04483>

329 [9] Microsoft. 2019. *Eye control troubleshooting guide*. Online. <https://support.microsoft.com/en-us/windows/eye-control-troubleshooting-guide-83352d28-688a-e29c-4b5f-e91aee324259>.

330 [10] Payod Panda, Molly Jane Nicholas, David Nguyen, Eyal Ofek, Michel Pahud, Sean Rintel, Mar Gonzalez-Franco, Ken Hinckley, and Jaron Lanier.
 331 2023. Beyond Audio: Towards a Design Space of Headphones as a Site for Interaction and Sensing. In *Proceedings of the 2023 ACM Designing
 332 Interactive Systems Conference* (Pittsburgh, PA, USA) (DIS '23). Association for Computing Machinery, New York, NY, USA, 904–916. <https://doi.org/10.1145/3563657.3596022>

333 [11] Rafael Raya, Javier O. Roa, Eduardo Rocon, Ramón Ceres, and José L. Pons. 2010. Wearable inertial mouse for children with physical and cognitive
 334 impairments. *Sensors and Actuators A: Physical* 162, 2 (2010), 248–259. <https://doi.org/10.1016/j.sna.2010.04.019> Eurosensors XXIII, 2009.

335 [12] Andreia Sias Rodrigues, Vinicius Kruger da Costa, Rafael Cunha Cardoso, Marcio Bender Machado, Marcelo Bender Machado, and Tatiana Aires
 336 Tavares. 2017. Evaluation of a Head-Tracking Pointing Device for Users with Motor Disabilities (PETRA '17). Association for Computing Machinery,
 337 New York, NY, USA, 156–162. <https://doi.org/10.1145/3056540.3056552>

338 [13] Dariusz Sawicki and Piotr Kowalczyk. 2018. Head Movement Based Interaction in Mobility. *International Journal of Human–Computer Interaction*
 339 34, 7 (2018), 653–665. <https://doi.org/10.1080/10447318.2017.1392078> arXiv:<https://doi.org/10.1080/10447318.2017.1392078>

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364