

ThumbStick: 片手スマートウォッチ操作のための 親指の二次元動作認識システム

青山 周平^{1,a)} 志築 文太郎^{2,b)}

概要: 本稿では、スマートウォッチにおいて二次元方向への入力を可能にするシステム「ThumbStick」を示す。ThumbStick は、スマートウォッチを装着した手の親指における、二次元方向の動きを認識するシステムである。本システムは、ユーザがスマートウォッチのタッチディスプレイに触れることなく、片手のみを用いた操作を行うことを可能にする。本システムの実装は、親指動作に伴う手首の形状変化を利用しており、その形状変化から親指の位置を識別する。システムは、親指の位置を識別することにより、ユーザがジョイスティックを扱うかのような入力を実現する。

ThumbStick: Two-dimensional Thumb Motion Recognition System for One-handed Smartwatch Input

AOYAMA SHUHEI^{1,a)} SHIZUKI BUNTAROU^{2,b)}

Abstract: In this paper, we show the system *ThumbStick*, which realizes a two-dimensional input on a smartwatch, with only the thumb of the hand wearing the smartwatch. The system estimates the thumb position using the change of the wrist's contour. Thus, the user can operate the smartwatch with a joystick-like two-dimensional input using the thumb.

1. はじめに

スマートウォッチとは、腕時計を模して作られた手首装着型の情報端末である。スマートウォッチの多くは、腕時計の文字盤にあたる部分にタッチディスプレイが組み込まれており、これが時刻の表示のみならず、様々な機能を提供することを可能としている。

しかし、タッチディスプレイを用いた入力は、スマートウォッチ操作において不都合となる場合がある。例えば、ユーザが画面を見ながら操作を行う際、指が画面を隠してしまい操作が行いづらくなる状況が発生し得る。また、

タッチディスプレイを用いた操作は、スマートウォッチ装着側とは逆側の手を用いてディスプレイに触れる必要がある。すなわち、スマートウォッチの片手操作が行えない。そのため、ユーザが荷物を持っている状態などにおいて、スマートウォッチを操作することを困難にする。

本研究では、上記の問題を解決する手段として、親指動作を用いたスマートウォッチ装着側の手による操作に着目する。親指動作をスマートウォッチへの入力に使うことは、タッチディスプレイを用いない片手操作を実現することのみならず、様々な点において利点がある。例えば、親指は他の指の状態に影響を受けないため、物などを掴んだ状態においても動作を行うことが可能な点である。これは、鞆を持った状態においてもスマートウォッチ入力を行うことを可能にする。また、操作に要する予備動作が小さいため、素早く操作に移行できるという利点がある。さらに、操作の際に必要な動作自体も小さいため「手が疲れにくい」「人目をばからずに操作が行える」という利点もある。

本研究では、親指動作を用いたスマートウォッチ操作を

¹ 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻

Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba

² 筑波大学システム情報系

Faculty of Engineering, Information and Systems, University of Tsukuba

a) aoyama@iplab.cs.tsukuba.ac.jp

b) shizuki@cs.tsukuba.ac.jp

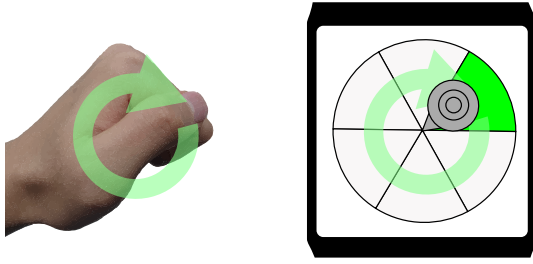


図 1 ThumbStick を用いたスマートウォッチの操作イメージ。
Fig. 1 Operation image of a smartwatch with ThumbStick.

実現するためのシステム「ThumbStick」を示す。ThumbStick はスマートウォッチベルトに取り付けた赤外線近接センサアレイを用いて、人差し指の側面上における親指の位置を識別することにより、親指動作を認識するシステムである。ユーザは本システムを用いることにより、親指の動きをジョイスティックの動きに見立てて、二次元方向への入力を行うことが可能となる（図 1）。本稿においては、製作したセンサデバイスを用いて親指位置を識別できるかを評価するために行った著者実験について述べるとともに、本システムの実装方法を示す。

2. 関連研究

本研究は手を握った際にできる人差し指側面平面上における親指の位置を識別することにより、親指動作を用いたスマートウォッチの片手操作を実現するための研究である。

2.1 スマートウォッチの片手操作に関する研究

本研究と同様に、スマートウォッチの片手操作に着目した研究は存在する。Gong らは、手首の動きを用いてスマートウォッチを片手で操作する手法 WristWhirl を示した [1]。Sun らは、スマートウォッチを傾けることで片手操作を行う手法 Float を示した [2]。Kerber らは、筋電位センサバンドを用いたスマートウォッチの片手操作が可能であることを調査した [3]。これらの研究は手首や腕、手全体の動きをスマートウォッチへの入力に用いる一方、本研究は親指の動きを入力に用いる。

また、親指を用いてスマートウォッチを片手操作することに着目した研究も存在する。福本らは手首に取り付けた端末の加速度センサを用いて、装着した手の親指と任意の指先とを叩き合わせる動作を検出し、このリズムパターンを用いて片手操作を行う方法を示した [4]。この指先同士の接触を入力とする福本らの研究に対し、本研究は親指の二次元方向への動きを入力に用いる。また、Huang らは、スマートウォッチを片手操作する方法として、親指で他の指の腹を触れる操作手法 DigitSpace を示した [5]。指の腹を入力面として利用する Huang らの研究に対して、本研究は手を握った際にできる人差し指側面を入力面として利用する。

2.2 手/指動作認識に関する研究

本研究と同様に、列状に多数並べたセンサ（センサアレイ）を手首に取り付けることにより手/指動作を認識する研究も存在する。Zhang らは電気インピーダンス・トモグラフィを用いた Tomo というシステムを開発し、手のジェスチャを認識した [6]。Rekimoto らは、手首に巻き付けた静電容量式のセンサアレイを用いて、指の動作に伴う腱や筋肉の動きから生じる手首の形状変化を計測することにより、手のジェスチャを認識した [7]。また、Dementyev らは、手首に感圧（圧力）センサアレイを巻きつけることにより同様のことを行った [8]。同様に、Fukui ら [9] および Ortega-Avila ら [10] は、手首に赤外線センサアレイを巻き、センサと皮膚間の距離から手首の形状変化を計測することにより、手のジェスチャを認識した。同様に Gong らは、手首に赤外線センサアレイを巻くことにより、手首の動きを認識した [1]。McIntosh らも同様に、赤外線センサアレイを備えたプレスレット型デバイス SensIR を開発し、手のジェスチャを認識した [11]。ただし、McIntosh らはセンサと皮膚間の距離を測るのではなく、赤外線が手首を透過することに着目した。これらの研究では、手全体の姿勢や動きを認識することを目指している一方、本研究は親指の動きのみに着目し、より小さなジェスチャ操作を実現することを目指している。

3. 親指動作の認識

本研究では、手を握った際にできる人差し指側面平面上における親指の動きを認識し、その動作をスマートウォッチへの入力とする。本節においては、親指動作を認識する原理を述べるとともに、その実装について述べる。

3.1 認識原理

親指動作を認識するにあたり、本研究は手首の形状変化を利用して親指の位置を識別する。手首の形状変化とは、指の動作に伴う筋肉の収縮や腱の移動により、手首の輪郭形状が変化することを指す。本研究は、この手首の形状変化をスマートウォッチベルトの内側に取り付けたセンサアレイを用いて測定し、親指位置ごとのセンサ値を学習することにより親指の位置を識別する。

3.2 手首形状測定デバイス

親指の位置を識別するにあたり、手首形状を測定するためのセンサデバイスを製作した。本デバイスは、マグネットバックルタイプのスマートウォッチベルトと一体化するように設計されている（図 2A）。このデバイスと皮膚とが接する面には、手首を覆うように赤外線近接センサアレイとそれを囲む枠が取り付けられている（図 2B）。本デバイスは、枠により手首の皮膚とセンサの間に隙間を設け、それぞれのセンサがこの隙間の距離を測ることにより手首の

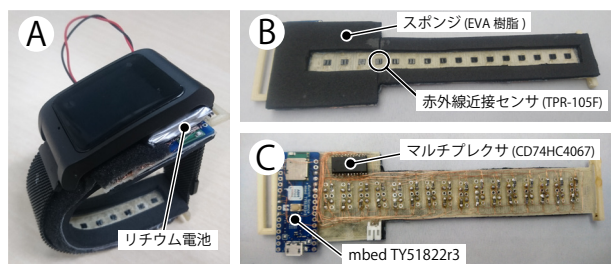


図 2 手首形状測定用のセンサデバイス：A) スマートウォッチベルトへの装着時，B) デバイスのセンサアレイ面，C) デバイス内部。

Fig. 2 Wrist-worn sensor device. A) Overview, B) the side of sensors array, C) the inside of the device.

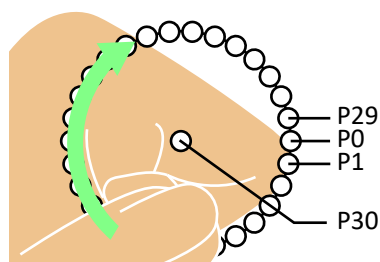


図 3 識別対象とする親指の位置。

Fig. 3 The thumb position to be identified.

形状を取得する。

本センサデバイスの構造について説明する。本デバイスは、柔軟性のあるフィラメント（PVA Filament 1.75 mm natural, Shenzhen Esun Industrial）を用いて 3D プリンタ（FLASHFORGE Dreamer, Apple Tree）から出力された厚さ 1 mm の土台上に構築されている（図 2C）。土台上には、マルチプレクサ（CD74HC4067, Texas Instruments）および 16 個の赤外線近接センサ（TPR-105F, GENIXTEK）が 7.68 mm 間隔にて取り付けられている。また、赤外線近接センサから得たセンサ値を送るため、BLE（nRF51822 Rev.3 SoC, Nordic Semiconductor）が組み込まれた mbed（mbed TY51822r3, Switch Science）が取り付けられている。なお、赤外線近接センサアレイを囲む枠は高さ 2 mm である。

4. 親指位置識別精度の評価

製作したセンサデバイスから得たセンサ値をもとに親指位置の識別が行えるかを評価するため、著者 1 名（男性，年齢 24 歳，右利き）を被験者とした実験を行った。本実験が識別の対象とするのは、手を握った際にできる人差し指側面の平面上における親指の位置である。ここで述べる親指の位置とは、図 3 のように親指を人差し指の側面で回転させるようになぞった際の円周上の各点（P0–P29）および円の中心（P30）のことを指す。以降、P0–P29 を円周点、P30 を中心点と呼称する。

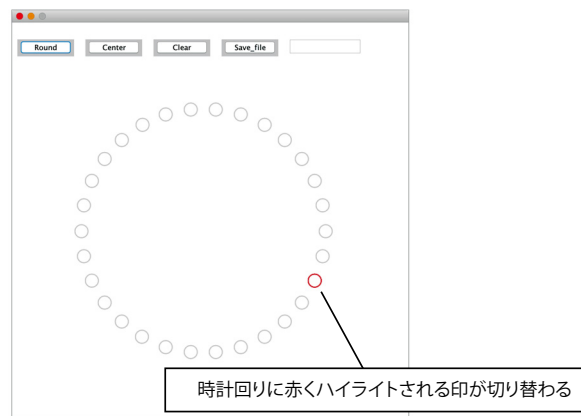


図 4 実験に使用したアプリケーション。

Fig. 4 Experimental application.

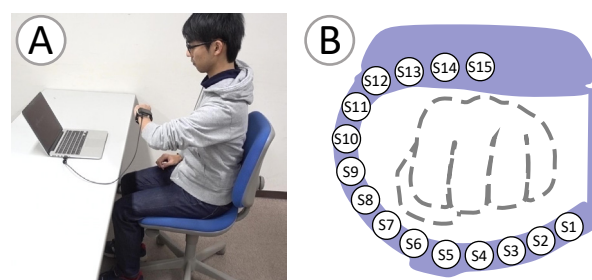


図 5 実験環境：A) 実験時の姿勢，B) センサデバイスにおける各センサの位置。

Fig. 5 Experiment environment: A) Posture, B) The position of each sensor in the sensor device.

4.1 実験アプリケーション

実験に使用したアプリケーションは Processing (ver 2.2.1) にて実装された。図 4 に本アプリケーションのスクリーンショットを示す。本アプリケーション画面には、円形状に「○」印が並べて表示されている。実験が開始されると、右端の印（時計における 3 時の位置の印）が赤くハイライトされる。この位置におけるセンサ値の記録が終わると、時計回りにハイライトされる印が切り替わる。被験者がこのハイライトされた印の位置に合わせて親指を動かすことで、アプリケーションは円周点におけるセンサ値を記録する。また、本アプリケーションは円周点におけるセンサ値の記録後、被験者が親指を中心点に動かし、ボタンを押すことにより、中心点におけるセンサ値を記録する。

4.2 実験環境

実験時の姿勢を図 5A に示す。本実験は、被験者が椅子に深く腰掛けた状態にて行われた。また、被験者の左手首にはセンサデバイスが取り付けられた。このセンサデバイスにおける各センサの位置を図 5B に示す。なお、本実験においては BLE による通信ではなく、通信速度がより速い USB ケーブルによる通信にてセンサ値を PC に転送した。

表 1 親指位置の識別結果の混同行列.

Table 1 Confusion matrix of classification result of thumb positions.

		識別された親指位置 (%)																														
実際の親指位置 (%)	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30	
	P0	20	60			10																								10		
	P1	50	20	20			10																									
	P2		20	30	40	10																										
	P3			50	10	30	10																									
	P4			10	20	10	30	20	10																							
	P5			10	10	30	20	30																								
	P6				10	20	20	20		10																						
	P7				10	10	20	20	10	20													10									
	P8					10	10	10	20	30	20																					
	P9							30	10	20	10	30																				
	P10								30	20		30	20																			
	P11									30	30		40																			
	P12								10	10	30		50																			
	P13									10	50			40																		
	P14												40		60																	
	P15													10	50		20	10	10													
	P16													30	50	10	20	10														
	P17													20	30	10	20	10			10											
	P18														10	20	30	20					10	10								
	P19															40	20	10	20		10	20										
	P20																	10	30	40		40			10							
	P21																			20	40	30	10									
	P22																				40	20	40									
	P23																10	10				20	20		40							
	P24																				10		20	20	40	10						
	P25																					10			30	20	40					
	P26																									20		70	10			
	P27																										60		40			
	P28	10	10																										30	20	30	
	P29	20	10																											60	10	
	P30																						10									90

4.3 実験手順

実験手順を説明する。まず、被験者はセンサデバイスを装着した左手及び腕を図 5A のように胸の前に構えた姿勢をとる。またこの際、左手は軽く握る。次に、被験者は実験アプリケーション上のボタンをマウスにてクリックし、円周点におけるセンサ値の記録を開始する。記録開始後、被験者は赤くハイライトされた「○」印の位置に合わせて、親指を移動させる。本実験においては、円周点におけるセンサ値の記録を 12 周分連続して行った (P29 の測定後すぐに P0 から記録を再開する)。なお、円周点を 1 周記録するのにかかる時間はおよそ 2.0 秒である。

円周点におけるセンサ値を記録後、被験者は中心点におけるセンサ値を記録する。被験者はまず、親指を中心点 (P30) に動かし、中心点記録用のボタンをクリックする。ボタンがクリックされた際、1 回分のセンサ値が記録される。その後、被験者は一度親指を P0 の位置に動かし、再度中心点に親指を動かしてボタンをクリックする。被験者はこの中心点におけるセンサ値の記録を、10 回繰り返す。

4.4 結果及び考察

実験から得たデータから親指位置の識別精度を評価するため、機械学習ソフトウェアである Weka Machine Learning Toolkit[12] を用いて、10-fold 交差検定を行った。交差検定に用いたデータは、円周点の記録における最初と最後の周を除いた 10 周分のデータ (10 周 × 30 点 × 16 次元のセンサ値データ) および中心点における 10 回分のデータ

(10 回 × 1 点 × 16 次元のセンサ値データ) である。また、分類器には LIBSVM[13] を使用した。なお、円周点の記録において最初と最後の周のデータを除いた理由は、動作の開始および終了時のデータを除くためである。

交差検定の結果を表 1 に示す。検定の結果、親指位置の識別率は 16.1% と低い識別率であった。しかし、各円周点の識別において大きな識別位置のずれはなく、いずれも付近の円周点との誤識別が起こっていた。これは、円周点におけるセンサ値の記録の際、周囲ごとに親指を動かす速度がずれることにより発生した可能性がある。なお、各円周点の識別において、ずれが 24 度以内に収まる確率 (すなわち、両隣を含めた 3 点のいずれかに識別される確率) は 76.8% であった。また、識別のずれが 48 度以内に収まる確率 (すなわち、両側 2 つ隣まで含めた 5 点のいずれかに識別される確率) は 91.3% であった。

以上の結果より、現状のセンサ値の記録 (学習) 方法においても、おおよそその親指位置の識別は可能であると考えられる。特に、中心点の識別率は 90% であり、中心点からどの方向に親指が動いたかを認識することが可能であることが示唆される。

5. ThumbStick

実験の結果をもとに、機械学習による親指位置推定に基づく親指動作認識システム「ThumbStick」を実装した。ThumbStick は親指の動きをジョイスティックの動きに見立てることにて、2 次元方向への入力を可能とするシステ

ムである。本節においては、ThumbStick のシステム構成およびシステムの使用手順について述べる。

5.1 システム構成

実装したシステムは、手首形状測定用のセンサデバイスおよび、親指動作認識処理用のスマートフォン（SONY Xperia Z5 E6653; Android 6.0.1）、画面表示用のスマートウォッチ（SONY SmartWatch 3 SWR50; Android Wear 1.5.0）から成る。センサデバイスから得られたセンサ値は BLE 通信を用いてスマートフォンに送られ、そこで親指動作認識にかかわる処理が行われる。また、スマートフォンにて処理された結果は、スマートウォッチに表示される。

5.2 親指位置の学習

ThumbStick は、実験時と同様に各円周点におけるセンサ値と中心点におけるセンサ値を学習させることにより、親指の位置を識別する。なお、本実装においては、機械学習ライブラリとして、LIBSVM[13] を用いた。

図 6A にスマートフォンに表示されるアプリケーション画面を示す。ユーザは実験と同様の手順により、親指位置の学習を行う（すなわち、円周点の学習後、中心点の学習を行う）。学習終了後、アプリケーションは親指位置の識別を行う。本アプリケーションにおいて、識別された円周点は緑色の「○」にて表示されている（図 6B）。またこの際、アプリケーションは識別時における尤度（0-1 の値にて表される）を重みとして、親指位置を推定する。すなわち、親指の位置（ X 座標および Y 座標）は以下のような式にて表される。

$$X = \sum_{i=P0}^{P29} \{Prob_i \times \cos(Rad_i)\} \times r \quad (-r < X < r)$$

$$Y = \sum_{i=P0}^{P29} \{Prob_i \times \sin(Rad_i)\} \times r \quad (-r < Y < r)$$

この式において、 $Prob_i$ は円周点 i における尤度を表し、 Rad_i は円周点 i における弧度を表す。また、 r は円の半径（中心点から各円周点までの距離）を表す。なお、この処理は、実験時における識別位置のずれを緩和する目的、および中心点と円周点間の親指位置を補完することを目的として行われる。本アプリケーションにおいて、推定された親指位置は赤色の「●」にて表示されている（図 6B）。

5.3 スマートウォッチ画面への表示

スマートフォンにて推定した親指位置を用いて、スマートウォッチ画面上に表示を行う。図 6C にスマートウォッチに表示されるアプリケーション画面を示す。スマートウォッチ画面には、親指位置をジョイスティックの位置に見立てたコントローラが表示されている。ユーザはこの仮想ジョイスティックを親指を動かして操作することにより、

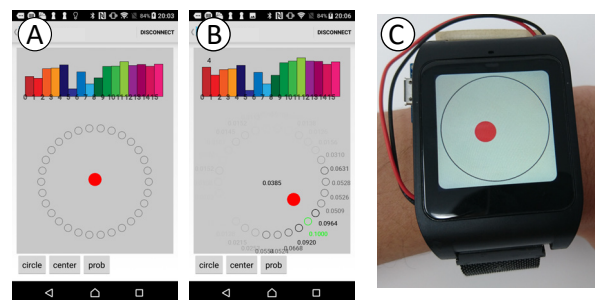


図 6 ThumbStick システムにおけるアプリケーション画面：A) スマートフォンアプリケーション-学習前, B) スマートフォンアプリケーション-学習後, C) スマートウォッチアプリケーション

Fig. 6 Application screenshot: A) Smartphone application - before training, B) smartphone application - after training, C) smartwatch application

スマートウォッチへの二次元入力を行うことができる。

6. 議論

本稿では、片手にてスマートウォッチを操作するための二次元入力手法を示した。しかし、現在のシステムにおいては有効な決定操作がないため、本システムにおいて操作できる範囲は限られてしまう。そのため、今後は親指を人差し指側面に押し込むような動作を学習させることにて、決定操作を実現することを考えている。

また、スマートウォッチは常に身に着けられているものであるため、日常生活において誤動作が起きないように、ThumbStick を起動する操作が必要である。今後は Float[2] のように、腕を上げて手首を回転させる動作を認識することにて、これを行うことを検討している。

7. まとめと今後の課題

本稿は、親指動作に伴う手首の形状変化を利用した、親指の二次元動作認識システムを示した。このシステムは手首に取り付けたセンサアレイから得たセンサ値をもとに、手を握った際にできる人差し指側面平面上における指の位置を識別し、親指動作を認識する。本稿では、手首の形状変化を測定するために製作したスマートウォッチベルト装着型センサデバイスを示した。また、製作したセンサデバイスを用いて親指位置を識別できるかを評価するために行った著者実験について述べた。本稿ではさらに、その実験結果をもとに作製した親指の二次元動作認識システムの実装について述べた。

今後は、本稿にて行った著者実験を他の被験者を対象として行い、本システムの適用範囲を検証していく。また、決定動作やシステムの起動動作を認識できるようにすることで、本システムの実用性を高めるつもりである。

参考文献

- [1] Gong, J., Yang, X.-D. and Irani, P.: WristWhirl: One-handed Continuous Smartwatch Input Using Wrist Gestures, in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pp. 861–872, New York, NY, USA (2016), ACM.
- [2] Sun, K., Wang, Y., Yu, C., Yan, Y., Wen, H. and Shi, Y.: Float: One-Handed and Touch-Free Target Selection on Smartwatches, in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pp. 692–704, New York, NY, USA (2017), ACM.
- [3] Kerber, F., Lessel, P. and Krüger, A.: Same-side Hand Interactions with Arm-placed Devices Using EMG, in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pp. 1367–1372, New York, NY, USA (2015), ACM.
- [4] 福本雅朗, 外村佳伸: 指釦: 手首装着型コマンド入力機構, 情報処理学会論文誌, Vol. 40, No. 2, pp. 389–398 (1999).
- [5] Huang, D.-Y., Chan, L., Yang, S., Wang, F., Liang, R.-H., Yang, D.-N., Hung, Y.-P. and Chen, B.-Y.: DigitSpace: Designing Thumb-to-Fingers Touch Interfaces for One-Handed and Eyes-Free Interactions, in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pp. 1526–1537, New York, NY, USA (2016), ACM.
- [6] Zhang, Y. and Harrison, C.: Tomo: Wearable, Low-Cost Electrical Impedance Tomography for Hand Gesture Recognition, in *Proceedings of the 28th Annual ACM Symposium on User Interface Software Technology*, UIST '15, pp. 167–173, New York, NY, USA (2015), ACM.
- [7] Rekimoto, J.: GestureWrist and GesturePad: Unobtrusive Wearable Interaction Devices, in *Proceedings of the 5th IEEE International Symposium on Wearable Computers*, ISWC '01, pp. 21–27, Washington, DC, USA (2001), IEEE Computer Society.
- [8] Dementyev, A. and Paradiso, J. A.: WristFlex: Low-power Gesture Input with Wrist-worn Pressure Sensors, in *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pp. 161–166, New York, NY, USA (2014), ACM.
- [9] Fukui, R., Watanabe, M., Shimosaka, M. and Sato, T.: Hand Shape Classification in Various Pronation Angles using a Wearable Wrist Contour Sensor, *Advanced Robotics*, Vol. 29, No. 1, pp. 3–11 (2015).
- [10] Ortega-Avila, S., Rakova, B., Sadi, S. and Mistry, P.: Non-invasive Optical Detection of Hand Gestures, in *Proceedings of the 6th Augmented Human International Conference*, AH '15, pp. 179–180, New York, NY, USA (2015), ACM.
- [11] McIntosh, J., Marzo, A. and Fraser, M.: SensIR: Detecting Hand Gestures with a Wearable Bracelet Using Infrared Transmission and Reflection, in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pp. 593–597, ACM (2017).
- [12] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. H.: The WEKA Data Mining Software: An Update, Vol. 11, pp. 10–18, New York, NY, USA (2009), ACM.
- [13] Chang, C.-C. and Lin, C.-J.: LIBSVM: A Library for Support Vector Machines, Vol. 2, pp. 27:1–27:27, New York, NY, USA (2011), ACM.