Centro Universitário de Brasília - UniCEUB
Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS
Engenharia de Computação

Autenticação em dispositivos móveis com acelerômetro usando biometria comportamental baseada em gestos

Autor: Matheus Bichara de Assumpção
Orientador: Professor Msc. Francisco Javier De Obaldía Díaz

Brasília, DF
2018

Matheus Bichara de Assumpção

Autenticação em dispositivos móveis com acelerômetro usando biometria comportamental baseada em gestos

Monografia submetida ao curso de graduação em Engenharia de Computação do Centro Universitário de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Centro Universitário de Brasília - UniCEUB

Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS

Orientador: Professor Msc. Francisco Javier De Obaldía Díaz

Brasília, DF

2018

Matheus Bichara de Assumpção

Autenticação em dispositivos móveis com acelerômetro usando biometria comportamental baseada em gestos/ Matheus Bichara de Assumpção. – Brasília, DF, 2018-

52 p. : il. (algumas color.) ; 30 cm.

Orientador: Professor Msc. Francisco Javier De Obaldía Díaz

Trabalho de Conclusão de Curso – Centro Universitário de Brasília - UniCEUB
Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS , 2018.

1. Autenticação biométrica. 2. Reconhecimento de gestos. I. Professor Msc. Francisco Javier De Obaldía Díaz. II. Centro Universitário de Brasília - UniCEUB. III. Faculdade de Tecnologia e Ciências Sociais Aplicadas. IV. Autenticação em dispositivos móveis com acelerômetro usando biometria comportamental baseada em gestos

CDU 02:141:005.6

Matheus Bichara de Assumpção

Autenticação em dispositivos móveis com acelerômetro usando biometria comportamental baseada em gestos

Monografia submetida ao curso de graduação em Engenharia de Computação do Centro Universitário de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Trabalho aprovado. Brasília, DF, novembro de 2018:

Professor Msc. Francisco Javier De Obaldía Díaz
Orientador

Professor Msc. Luís Cláudio Lopes de Araújo
Convidado 1

Professor Msc. Ivandro da Silva Ribeiro
Convidado 2

Brasília, DF
2018

Agradecimentos

Agradeço aos meus pais, meu irmão e minha família que, com muito carinho e amor, estiveram presentes em mais esta etapa da minha vida, como meus maiores incentivadores.

Agradeço a todos os professores do curso, que foram tão importantes na minha vida acadêmica.

Agradeço aos amigos e colegas do curso, pelo convívio, pela compreensão e pela amizade.

Resumo

Segurança em dispositivos móveis tornou-se uma grande preocupação nos últimos anos. Hoje em dia, as pessoas carregam a maioria de suas informações pessoais dentro de um smartphone: contas bancárias, números de cartão de crédito, contatos, dados de mídia social, senhas e muito mais. Se o dispositivo for perdido ou roubado, isso pode resultar em sérios danos financeiros e pessoais. Os mecanismos de autenticação atuais implantados em dispositivos móveis para proteção contra perda ou roubo geralmente usam um número PIN ou um padrão de desbloqueio. Esses métodos, além de não serem tão seguros, não são convenientes do ponto de vista do usuário, que geralmente desistem de usá-lo depois de algum período. Algumas abordagens mais recentes usam informações biométricas, como reconhecimento de digitais e facial, porém atualmente não estão disponíveis para todos os dispositivos, ou então exigem hardware interno especial. Como uma alternativa a essas soluções existentes, é proposto um método de autenticação para dispositivos móveis que usa gestos de movimento, baseados em acelerômetro, como biometria comportamental. O desempenho do sistema e as taxas de erro foram avaliados. Como os sensores de acelerômetro estão presentes em praticamente todos os smartphones, esse sistema de autenticação pode ser implantado basicamente em todos os dispositivos móveis.

Palavras-chaves: Autenticação biométrica, Reconhecimento de gestos, Sensor acelerômetro, Modelos Ocultos de Markov.

Abstract

Security on mobile devices have become a major concern over the past few years. Nowadays, an average person carries most of his personal information inside a smartphone: bank accounts, credit card numbers, contacts, social media data, passwords and more. If the device is lost or stolen, it can result in serious financial and personal damage. The current authentication mechanisms deployed on mobile devices for protection against loss or theft generally use a PIN number or a swipe pattern. These methods, besides not being so secure, are not convenient from the user's perspective, who usually give up using it after some period. Some newer approaches use biometric information, such as fingerprint and face recognition, but these are currently not available for all devices or require special built-in hardware. As an alternative to these existing solutions, an authentication method for mobile devices which uses accelerometer-based motion hand gestures as a behavioral biometric is proposed. The system's performance and error rates were evaluated. As accelerometer sensors are present in basically every smartphone, this authentication system could be deployed on basically every mobile device.

Key-words: Biometric Authentication, Gesture Recognition, Accerelometer, Hidden Markov Model.

Lista de ilustrações

Lista de abreviaturas e siglas

CER	Crossover Error Rate - Taxa de Intersecção de Erros
CMC	Cumulative Matching Characteristics - Característica Cumulativa de Correspondência
EER	Equal Error Rate - Taxa de Erro Igual
FAR	False Acceptance Rate - Taxas de Falsas Aceitações
FRR	False Rejection Rate - Taxa de Falsas Rejeições
GAR	Genuine Acceptance Rate - Taxa de Aceitações Verdadeiras
HMM	Hidden Markov Model - Cadeia Oculta de Marlov
ROC	Receiver Operating Characteristics - Característica de Operação do Receptor
SI	Sistema de Informação

Sumário

1 Introdução

1.1 Motivação

O uso da biometria como mecanismo de autenticação aumentou consideravelmente nos últimos anos, uma vez que esta fornece maior segurança do que os métodos tradicionais de reconhecimento pessoal (PRABHAKAR; PANKANTI; JAIN, 2003). As aplicações vão desde imigração e controle de fronteiras, controle de acesso físico, aplicação da lei e da justiça e várias outras.

A biometria pode ser classificada em dois tipos: física ou comportamental. O primeiro tipo refere-se a um traço fisiológico ou intrínseco, por exemplo. a impressão digital ou rosto. Biometrias comportamentais, em contrapartida, são o resultado de padrões de atividade aprendidos (MAYRON, 2015), por exemplo, assinatura, voz, perfil de utilização de serviço, padrão de digitação (PEACOCK; KE; WILKERSON, 2004), maneira de andar (JORGENSEN; YU, 2011), dentre outros.

Nos últimos anos, tem havido um interesse crescente no uso da autenticação biométrica em uma determinada área: dispositivos móveis. Assim como os Smartphones se tornaram itens essenciais na vida da maioria das pessoas, as preocupações de segurança também estão crescendo.

Exemplos de sistemas de autenticação biométrica relativamente recentes em smartphones são o desbloqueio de impressões digitais Touch ID do iPhone 5S e o desbloqueio facial do Android 4.0, que estão entre os primeiros implantações em larga escala de biometria para consumidores (BHAGAVATULA et al., 2015).

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é propor, implementar e validar um sistema biométrico de autenticação alternativo para dispositivos móveis baseado em reconhecimento de gestos manuais a partir do uso de sensor acelerômetro.

1.2.2 Objetivos Específicos

- Levantar o referencial teórico relacionado a biometria e sistemas biométricos, problemas de autenticação e identificação, sensores do tipo acelerômetro e técnicas de reconhecimento de padrões aplicadas a gestos;

- Propor e implementar sistema de autenticação biométrico de gestos manuais, utilizando dados de sensor acelerômetro, e algoritmos de reconhecimento em software MATLAB;
- Levantamento em software MATLAB de parâmetros e estatísticas de falsos positivos e falsos negativos para análise de viabilidade da solução proposta;
- Implementação de aplicativo para sistema operacional Android como protótipo do sistema proposto.

1.3 Justificativa e Relevância do Tema

Conforme descrito em Mayron (2015), os dispositivos móveis desafiam a usabilidade de senhas de texto, pois elas dependem do manuseio com os dedos de uma tela sensível ao toque, o que retarda o processo de autenticação e torna a escolha de senhas mais fortes inconveniente sobre a perspectiva do usuário. O estudo realizado por Trewin et al. (2012) examinou o esforço do usuário e as taxas de erro de três modalidades de autenticação biométrica - voz, rosto e gesto - bem como a entrada de senha em um dispositivo móvel.

A maioria dos dispositivos móveis geralmente só permitem dois tipos de autenticação de usuário prontos para uso: PIN ou padrão de desbloqueio. Usando a primeira opção, o usuário inicialmente registra um número de quatro dígitos e é solicitado a inserir novamente o mesmo PIN sempre que desbloquear o smartphone. Na segunda abordagem, o usuário primeiro traça um padrão em uma grade de nove nós, que é gravado e usado posteriormente para desbloqueio. Uma grande desvantagem dessas abordagens é a falta de conveniência.

Ambos os métodos exigem que o usuário foque e concentre-se na tela do dispositivo por alguns segundos. Pode não parece muito à primeira vista, mas estudos mostraram que um usuário normal desbloqueia seu smartphone mais de cem vezes por dia em média. Por esse motivo, alguns usuários desativam esses mecanismos de autenticação após algum período de tempo.

O presente trabalho propõe uma abordagem de autenticação para dispositivos móveis que usa gestos manuais baseados em acelerômetro como biometria comportamental. Hipotetiza-se que um usuário pode proteger um dispositivo gravando primeiramente um gesto manual personalizado, por ex. uma assinatura no ar, e mais tarde, quando solicitado, repetir o mesmo gesto para ser autenticado.

A principal vantagem deste método é que o usuário não precisa se concentrar na tela do dispositivo enquanto executa o gesto de mão. Isso torna essa abordagem muito

mais conveniente, permitindo que o usuário desbloqueie facilmente seu smartphone em movimento, por exemplo, enquanto caminha.

1.4 Trabalhos Correlatos

O reconhecimento de gestos baseado em sensores acelerômetros foi abordado em várias publicações, embora não especificamente no que diz respeito ao problema de autenticação. No trabalho de Farella et al. (2006) investigou-se a viabilidade de se usar gestos de movimento como biometria, e os resultados mostraram alta porcentagem de correspondência. Um sistema de autenticação baseado em gestos para terminais públicos não confiáveis foi também proposto por Patel, Pierce e Abowd (2004).

Wang e Chuang (2012) propuseram um sistema de caneta digital baseado em acelerômetro que realiza reconhecimento de gestos. O artigo de Xu, Zhou e Li (2012) apresenta um sistema de reconhecimento de gestos de mão baseado em um sensor acelerômetro, mas depende de um dicionário de gestos pré-especificado, o que não é viável em um cenário de autenticação. Os trabalhos de Mäntyjärvi et al. (2004), Schlömer et al. (2008) e Kela et al. (2006) apresentam sistemas de reconhecimento de gestos que usam métodos estatísticos como Cadeias Ocultas de Markov (HMM) como seu algoritmo central.

Liu et al. (2009) propuseram um promissor algoritmo de reconhecimento, usando Dynamic Time Warping (DTW) para gestos de movimento, que requer uma única amostra de treinamento, ao contrário de métodos estatísticos, que requerem várias amostras de treinamento. O trabalho de Wu et al. (2009) apresenta uma abordagem de reconhecimento de gestos denominada FDSVM (Descritor baseado em quadros e SVM multi-classe), e compararam os resultados com os métodos existentes (HMM e DTW).

Neste projeto, a técnica Hidden Markov Model (HMM) foi escolhida, pois esta é conhecida por seu grande sucesso em reconhecimento de fala (RABINER, 1989) e também por ter sido implementada com sucesso em sistemas gestuais usando acelerômetro 3D. (SCHLöMER et al., 2008) (PYLVÄNÄINEN, 2005).

1.5 Resultados Esperados

Espera-se como resultados deste trabalho: implementar e validar uma solução de autenticação para dispositivos móveis baseada em reconhecimento de gestos manuais com o uso de sensor acerelômetro; desenvolver aplicativo protótipo e avaliar a viabilidade da solução proposta com base em métricas usadas pela literatura.

1.6 Estrutura da monografia

Esta monografia encontra-se estruturada da seguinte forma: este primeiro capítulo trata da contextualização do tema, aponta os objetivos geral e específicos do trabalho, justificativa e os trabalhos correlatos; no segundo capítulo, será apresentada a fundamentação teórica necessária ao desenvolvimento do projeto, sendo descritos os conceitos relacionados a biometrias e sistemas biométricos, reconhecimento de padrões e sensores do tipo acelerômetro; a descrição da solução proposta será detalhada no terceiro capítulo; os resultados e análises serão apresentados no quarto capítulo; por fim, no quinto capítulo, serão apresentadas as conclusões deste trabalho.

2 Referencial Teórico

2.1 Sistemas biométricos e biometria

Um sistema biométrico tem a finalidade de verificar se uma amostra pertence a um indivíduo conhecido ou não conhecido. No primeiro caso, os sistemas são denominados de identificação positiva e rejeitam o indivíduo que forneça uma amostra que não corresponda a uma das amostras guardadas. No segundo caso, os sistemas são denominados de sistemas de identificação negativa e rejeitam o indivíduo que forneça uma amostra que corresponda a uma das amostras guardadas. Deste modo, um sistema pode verificar se o indivíduo é quem reivindica ser ou verificar que realmente é desconhecido (WAYMAN et al., 2005).

O problema de estabelecer uma associação entre um indivíduo e uma identidade pode ser dividido em duas categorias: autenticação e identificação. Autenticação refere-se ao problema de confirmar ou negar identidade de um indivíduo, enquanto identificação se preocupa em estabelecer a identidade, desconhecida à priori, de um indivíduo (MAGALHÃES; SANTOS, 2003).

Uma das finalidades da autenticação é a de verificação, ou seja, determinar se um indivíduo é realmente quem ele alega ser (DELAC; GRGIC, 2004). A verificação é normalmente utilizada em procedimentos de identificação positiva. Para tanto, o usuário do sistema apresenta o elemento de autenticação, o qual vai ser comparado com o registo existente numa base de dados, previamente criada com todos os elementos de autenticação de cada um dos usuários envolvidos.

Assim, a verificação é efetuada através da comparação de um-para-um, sendo possível verificar se a característica apresentada pelo usuário/utilizador do sistema corresponde realmente ao indivíduo que afirma ser (JAIN; ROSS; PRABHAKAR, 2004). De igual forma, a autenticação biométrica tem a finalidade de identificar o indivíduo que pretende utilizar o sistema. Tal procedimento é realizado a partir do elemento de autenticação apresentado pelo usuário (características), o qual vai ser analisado e comparado com os elementos armazenados na base de dados, ou seja, o sistema efetua a comparação de um-para-muitos.

Um sistema biométrico é constituído por quatro componentes: módulo sensor, módulo de extração de característica, módulo de correspondência e o módulo de tomada de decisão (WAYMAN et al., 2005). Assim, os sistemas que utilizam as biometrias como forma de autenticação são constituídos por quatro importantes fases, quais sejam: a captura, a extração de características, a comparação e a decisão.

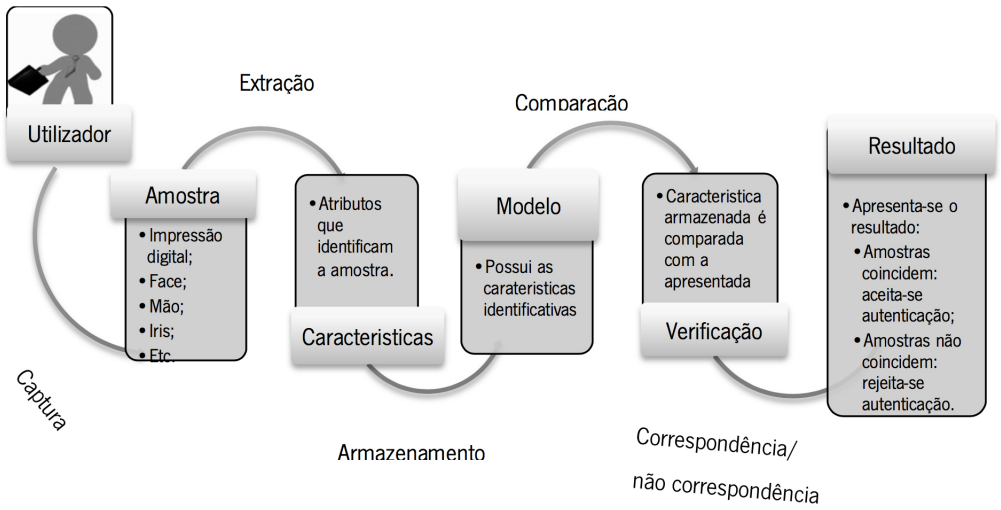
Na fase de captura obtém-se o elemento de autenticação, tipicamente sua repre-

sentação digital. Na segunda etapa faz-se a extração da informação e caraterísticas únicas para um modelo (ou padrão), que é posteriormente guardado. Após a criação e armazenamento do modelo é então possível fazer a comparação entre este e a amostra apresentada no momento de autenticação.

Segundo Delac e Grgic (2004), o módulo de correspondência efetua a comparação entre as características extraídas do utilizador do sistema e os modelos guardados, de forma a gerar scores. O resultado da identificação é uma lista dos indivíduos mais prováveis, ordenados segundo o score obtido na comparação.

Por fim o sistema vai negar ou aceitar a autenticação de acordo com a concordância entre o modelo guardado e a amostra apresentada. A figura 1 condensa as etapas atrás definidas, assim como o seu fluxo.

Figura 1 – Etapas de um sistema biométrico.



Fonte: (COSTA; OBELHEIRO; FRAGA, 2006)

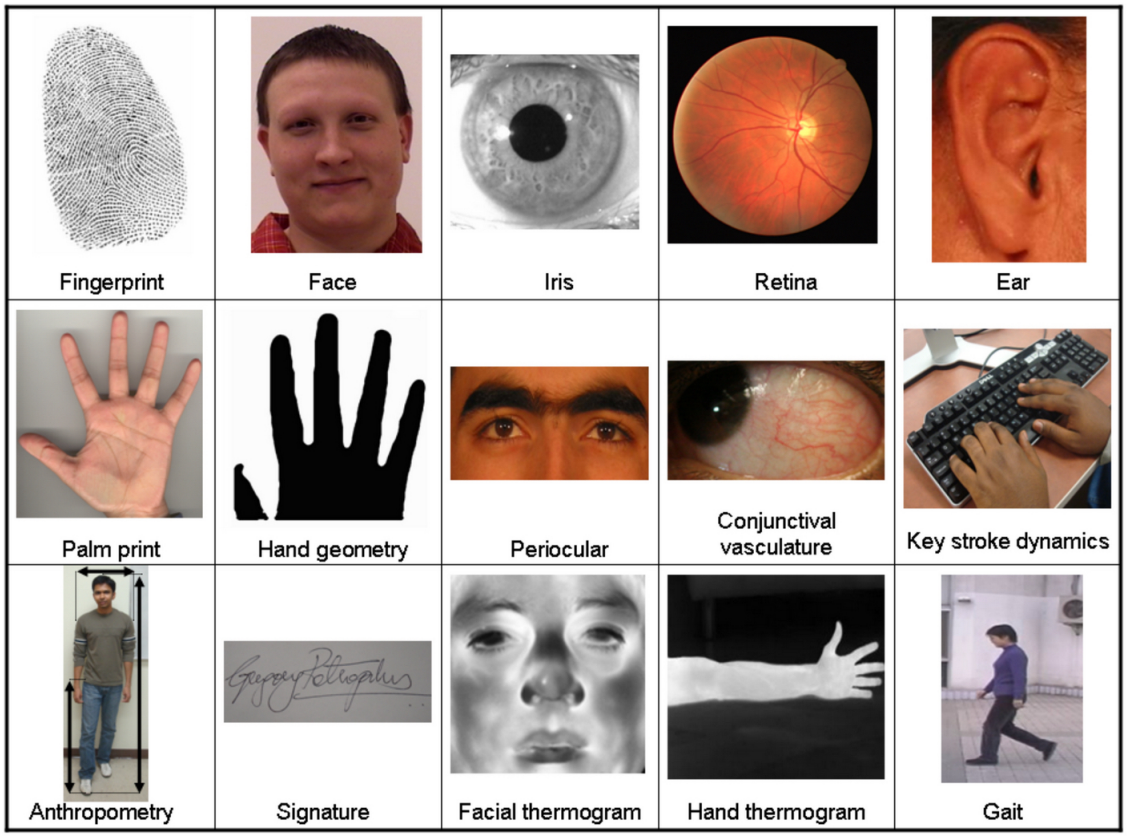
O termo biometria deriva do grego bios (vida) e metron (medida) e, na autenticação, refere-se à utilização de características próprias de um indivíduo para proceder à sua autenticação e/ou identificação perante um Sistema de Informação - SI de uma organização (MAGALHÃES; SANTOS, 2003).

A biometria pode ser classificada em duas categorias: fisiológicas e comportamentais. A biometria fisiológica diz respeito às caraterísticas físicas do indivíduo, que variam pouco ao longo do tempo (COSTA; OBELHEIRO; FRAGA, 2006). Cita-se com exemplo, as impressões digitais, reconhecimento facial, termograma facial, geometria da mão ou orelha, reconhecimento da iris ou da retina, entre outras. As biometrias comportamentais, por sua vez, referem-se ao comportamento do indivíduo. Estas são adquiridas ao passar do tempo, vez que vão sendo aprendidas ou desenvolvidas devido à sua constante utilização. Por ser consideradas dinâmicas alteram-se tanto por vontade própria como

pelo estado emocional do indivíduo (COSTA; OBELHEIRO; FRAGA, 2006). Alguns exemplos de biometrias comportamentais são a forma como se escreve a assinatura, a voz, perfil de utilização de serviço, padrão de digitação ou modo de andar (CLARKE; FURNELL, 2005).

A figura 2 mostra um conjunto de vários traços biométricos, fisiológicos e comportamentais.

Figura 2 – Conjunto de traços biométricos comuns.



Fonte: (JAIN; ROSS; PRABHAKAR, 2004)

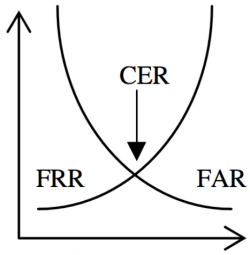
2.2 Análise de desempenho

Existem hoje muitas características utilizadas, isoladamente ou em conjunto, para autenticar e/ou identificar um indivíduo. Cada um dos métodos pode ser avaliado por meio de vários parâmetros: grau de fiabilidade, nível de conforto, nível de aceitação e custo de implementação (LIU; SILVERMAN, 2001).

O grau de fiabilidade pode ser aferido tendo em atenção os valores FAR (False Acceptance Rate – Taxa de Falsas Aceitações) e o FRR (False Rejection Rate – Taxa de Falsas Rejeições). Estes indicadores variam de forma inversa, em função do ponto de decisão estabelecido como o limiar do resultado da comparação entre o padrão apresentado e o que se encontra armazenado.

Por essa razão, infelizmente estas variáveis são mutuamente dependentes, não sendo possível minimizar ambas. Assim, procura-se o ponto de equilíbrio (fig. 3), o qual se chama CER (Crossover Error Rate – Taxa de Intersecção de Erros). Quanto mais baixo for o CER, mais preciso é um sistema biométrico (LIU; SILVERMAN, 2001). Este indicador de desempenho, o ponto em que FRR = FAR, também é designado por EER (Equal Error Rate).

Figura 3 – Distribuições FAR e FRR.



Fonte: (MAGALHÃES; SANTOS, 2003)

Assim, se o valor de limiar for escolhido para obter uma taxa baixa de Falsos Positivos, o sistema é mais seguro, pois é mais complicado para um impostor conseguir penetrar o sistema sem autorização, contudo tem a desvantagem de não permitir o acesso a alguns usuários com autorização, vez que a taxa de Falsos Negativos é elevada.

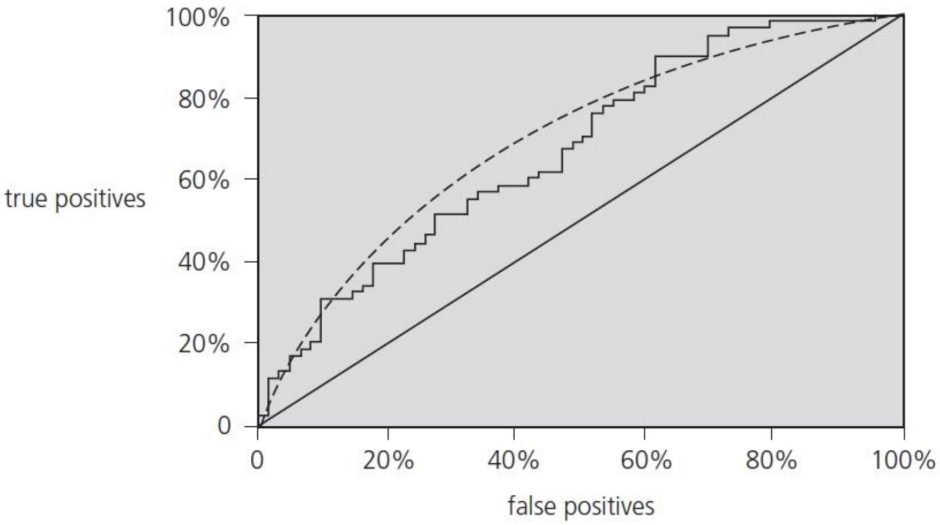
Por outro lado, caso haja taxa de Falsos Positivos elevada, o sistema é mais vulnerável, vez que permite o acesso a alguns indivíduos não autorizados, mas será menos impeditivo para os utilizadores legítimos do sistema (DELAC; GRGIC, 2004).

Nesse sentido, taxa de falsos positivos elevadas não são adequadas a sistemas que requerem um elevado grau de segurança. No entanto, tais sistemas podem ser utilizados em situações em que é necessário a identificação de um maior número de indivíduos. Um caso em que é adequado a utilização deste tipo de sistema é a análise forense (DELAC; GRGIC, 2004).

Outra forma de avaliar o desempenho é o Receiver Operating Characteristics (ROC). Deste gráfico é possível obter os valores de FRR, FAR e EER (HANLEY; MCNEIL, 1982). A figura 4 mostra alguns exemplos de curvas ROC. Este é um gráfico que expressa a taxa de verdadeiros positivos (eixo vertical) contra a taxa de falsos positivos (eixo horizontal).

Segundo Hanley e McNeil (1982), a curva ROC deve ser a mais próxima possível do canto esquerdo do gráfico, ou seja, quanto mais próxima se encontrar do canto superior esquerdo, melhor é o desempenho do sistema. É desejável que se tenha a menor taxa de falsos positivos enquanto se mantém uma alta taxa de verdadeiros positivos. Assim, idealmente um sistema biométrico perfeito possui uma Equal Error Rate de 0%, onde a FAR

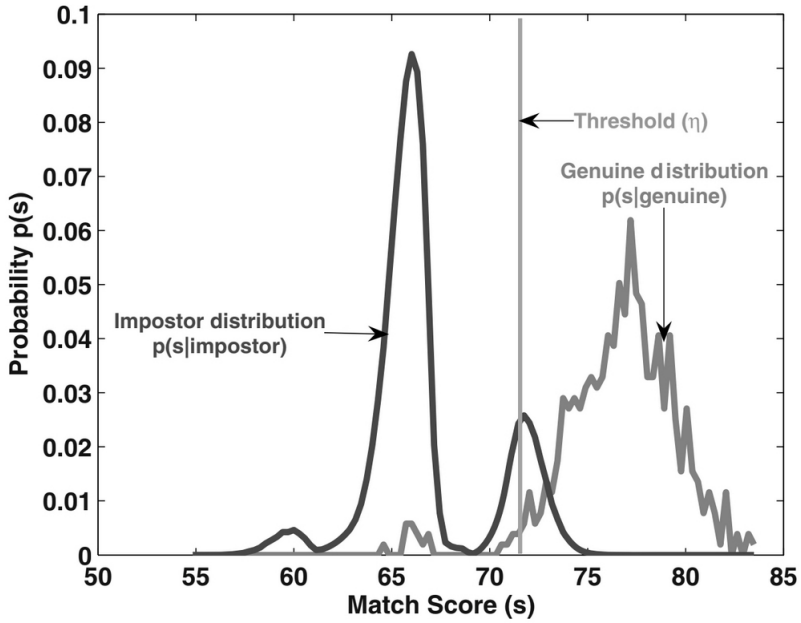
Figura 4 – Exemplos de curvas ROC.



Fonte: (DOMINGOS, 2014)

e FRR equivalem a 0%. Na curva ROC, é usualmente preferível comparar a taxa de falsos positivos FAR com a taxa de verdadeiros positivos, também chamada de GAR(Genuine Acceptance Rate), que equivale ao inverso da FRR.

Figura 5 – As distribuições de score genuína e impostora de uma sistema biométrico.



Fonte: (JAIN; ROSS; PRABHAKAR, 2004)

O gráfico ROC é obtido ao se variar o valor de limiar do score obtido, e calcular a FAR e a GAR/FRR para cada um desses valores. A figura 5 mostra as distribuições impostora e genuína para determinado sistema biométrico. Nota-se que, dadas estas duas

distribuições de score, a FAR e a FRR não podem ser reduzidas simultaneamente ao se ajustar o limiar. Assim, deseja-se que essas distribuições tenham a menor sobreposição possível.

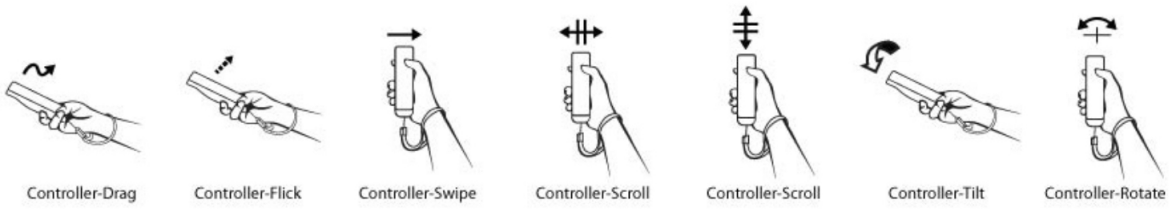
2.3 Reconhecimento de gestos manuais

O termo Reconhecimento de gestos refere-se coletivamente a todo o processo do seguimento dos gestos humanos para a sua representação e conversão em comandos com significado semântico (RAUTARAY; AGRAWAL, 2015).

Os gestos são movimentos significativos do corpo que envolvem movimentos físicos dos dedos, mãos, braços, cabeça ou face com a intenção de: (i) transmitir informações significativas ou (ii) interagir com o ambiente.

Dentre esses, os gestos manuais são muitas vezes os mais expressivos e os mais utilizados. Estes podem envolver: a) uma postura, configuração estática dos dedos sem movimento da mão; b) um gesto, o movimento dinâmico da mão, com ou sem movimento dos dedos.

Figura 6 – Exemplo de controle de *videogame* que utiliza reconhecimento de gestos.



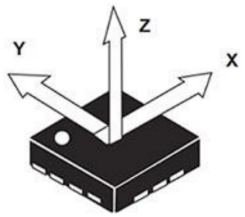
O objetivo deste trabalho é realizar o reconhecimento de gestos manuais dinâmicos com o uso de um sensor do tipo acelerômetro. A fig. 6 mostra um controle para jogos eletrônicos baseado nesse tipo de reconhecimento.

2.3.1 Sensor acelerômetro triaxial

Acelerômetros são dispositivos que medem a aceleração, que é a taxa de variação da velocidade de um objeto. Eles medem em metros por segundo ao quadrado (m/s^2) ou em forças G (g). Uma única força G equivale a aproximadamente $9,8m/s^2$, mas isso varia ligeiramente com a elevação. Os acelerômetros são úteis para detectar vibrações em sistemas ou para aplicações de orientação.

Acelerômetros são dispositivos eletromecânicos que detectam forças estáticas ou dinâmicas de aceleração. Forças estáticas incluem gravidade, enquanto forças dinâmicas podem incluir vibrações e movimento. Acelerômetros podem medir a aceleração em um, dois ou três eixos. Unidades de três eixos estão se tornando mais comuns à medida que o custo de desenvolvimento diminui.

Figura 7 – Representação de acelerômetro de três eixos comumente encontrados nos *smartphones* atuais.



Geralmente, os acelerômetros contêm placas capacitivas internamente. Algumas delas são fixas, enquanto outras são ligadas a minúsculas molas que se movem internamente à medida que as forças de aceleração atuam sobre o sensor. Conforme essas placas se movem em relação umas às outras, a capacitância entre elas muda. A partir dessas mudanças na capacitância, a aceleração pode ser determinada.

Outros acelerômetros podem ser centralizados em torno de materiais piezelétricos. Essas minúsculas estruturas de cristal geram carga elétrica quando colocadas sob estresse mecânico (por exemplo, aceleração).

2.4 Processo de Classificação

Ao passo que o reconhecimento de gestos estáticos geralmente pode ser feito por modelos de matching ou baseado em redes neurais, o problema de reconhecimento de gestos dinâmicos envolve o uso de técnicas como Hidden Markov model, Dynamic time warping, Time delay neural networks, Finite state machine, dentre outras (CÁRDENAS, 2015).

Neste projeto, a técnica Hidden Markov Model (HMM) ou Modelos Ocultos de Markov foi escolhida, pois esta é conhecida por seu grande sucesso em reconhecimento de fala (RABINER, 1989) e também por ter sido implementada com sucesso em sistemas gestuais usando acelerômetro 3D (SCHLöMER et al., 2008) (PYLVÄNÄINEN, 2005).

2.4.1 Modelos Ocultos de Markov

Um processo de Markov é um processo estocástico onde as distribuições de probabilidade para o seu desenvolvimento futuro dependem somente do estado presente, sem considerar como o processo chegou a tal estado (GRIGOLETTI, 2011).

Existem processos de Markov que são modelados como aproximações do mundo real, onde nem todos os estados são perfeitamente conhecidos. Esses modelos são denominados modelos ocultos de Markov (HMM), e a questão central em torno desses modelos é o grau com que são capazes de capturar a essência do processo escondido sob eles. Assim como as Cadeias de Markov (Markov Chains), os Modelos Ocultos de Markov (Hidden

Markov Models - HMM) apresentam uma série de vantagens na modelagem de sistemas com temporalidade inerente, onde um evento que acontece no instante atual é um desdobramento do evento que ocorreu no instante anterior (SILVA, 2015).

Os modelos de Markov escondidos, que surgiram originalmente no domínio de reconhecimento da fala, atualmente tem sido empregado como modelos de computação natural, em trabalhos sobre visão computacional e reconhecimento de manuscritos, de formas, de gestos e expressões faciais, em biologia computacional, entre outros (GRIGOLETTI, 2011).

Modelos ocultos de Markov são utilizados para processos Markovianos que geram observáveis de forma indireta, de acordo com as transições entre os estados da cadeia de Markov que governam o processo, mas que não podem ser diretamente observadas. Em outras palavras, a progressão do modelo está escondida do observador e a observação é indireta, feita por interferência, pois os observáveis são funções probabilísticas dos estados da cadeia ou das transições entre esses estados. Dessa forma, não é possível saber exatamente qual caminho ou sequência que levaram a determinada observação (ESPINDOLA, 2009).

Segundo Rabiner (1989), uma HMM é descrita de forma simbólica pela equação 2.1.

$$\lambda = (A, B, \pi) \tag{2.1}$$

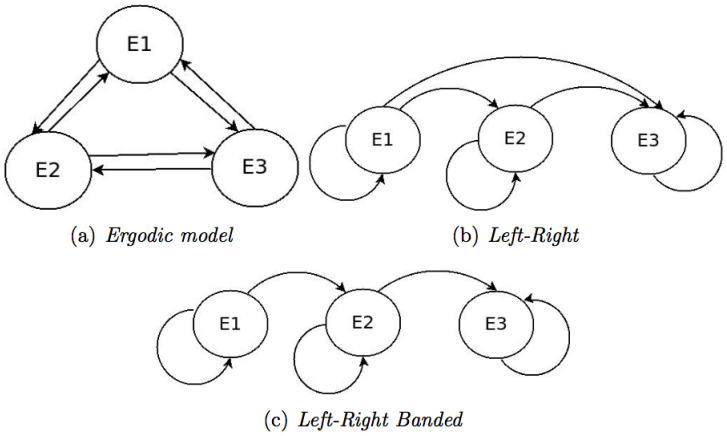
É caracterizada pelos elementos:

- $E = E_1, E_2, ..., E_N$: Conjunto de N estados;
- $S = S_1, S_2, ..., S_M$: Conjunto de M símbolos observáveis por estado;
- $A = [a_{ij}]_{N \times M}$: Matriz de probabilidades de transições de estados.
- $B = [b_{ij}]_{N \times M}$: Matriz de probabilidades de observação de símbolos.
- $\pi = [\pi_i]_N$: Vetor de distribuição inicial de estados.

Existem três tipos de topologia na hora da modelagem de HMMs, que podem ser vistos na figura 8.

Para que os gestos possam ser identificados é preciso que o sistema solucione três problemas: qual a probabilidade de um gesto pertencer a um determinado modelo oculto de Markov (HMM), se um gesto pertence a uma HMM qual a sequência de estados na HMM que tem a maior chance de produzi-lo e por último como estimar os parâmetros de

Figura 8 – Modelos de topologias HMM.



(a) *Ergodic model* (b) *Left-Right*

(c) *Left-Right Banded*

Fonte: (RABINER, 1989)

uma HMM de forma que ela apresente a melhor probabilidade de produzir determinados tipos de gestos (OLIVEIRA; MORITA, 2000).

Os dois primeiros problemas podem ser resolvidos pelo algoritmo de Viterbi, o terceiro problema pode ser solucionado utilizando-se do algoritmo de Baum-Welch.

2.4.1.1 Algoritmo de Virtebi

O Algoritmo de Virtebi é usado na resolução do problema de encontrar a sequência ótima de estados associada à sequência de observáveis.

Segundo Rabiner (1989), é difícil decidir um critério de otimização, dentre vários que possam existir. Dessa forma, a resolução desse problema poderia se dar de diferentes formas e diferentes sequências supostamente ótimas, de acordo com o critério de otimização ao escolhido.

Assim, o algoritmo em tela é uma solução recursiva para estimar a sequência de estados em um processo de Markov de estado finito e tempo discreto. Encontrar a sequência de estados mais provável, $E = E_1, E_2, ..., E_T$, dada uma sequência de símbolos $S = S_1, S_2, ..., S_T$, ou seja, maximizar $P(E|S, \lambda)$ equivale a maximizar $P(E, S|\lambda)$. Ambas operações vão devolver a sequência mais provável de estados.

Esse algoritmo funciona multiplicando os pesos das arestas anterior e próxima ao estado atual, obtendo assim sequências parciais mais prováveis. Uma escolha arbitrária acontece quando duas sequências apresentam a mesma probabilidade (ESPINDOLA, 2009).

2.4.1.2 Algoritmo de Baum-Welch

O algoritmo de Baum-Welch resolve o problema de estimar os parâmetros de uma HMM de tal forma que ela apresente a melhor probabilidade de produzir determinados tipos de sequências.

De acordo com Rabiner (1989), esse é o problema mais difícil dentre os três problemas fundamentais, porque não existe método analítico que permita que os paramêtros $\lambda = (A, B, \pi)$ que maximizam a probabilidade de um modelo gerar uma sequência completa de símbolos observáveis, $P(S|\lambda)$.

Contudo, o algoritmo de Baum-Welch resolve esse problema maximizando a probabilidade local. Diversos somatórios são realizados sobre o tempo de observação, T, com o intuito de se obter estimativas do número de vezes que cada estado é visitado em todo esse período.

2.4.2 Algoritmo K-means

O algoritmo de agrupamento K-Means (também chamado de K-Médias), proposto por MacQueen et al. (1967), é um método que tem como objetivo a partição de um conjunto de N observações em K grupos (ou clusters), onde cada observação corresponde ao grupo mais próximo. É um método muito utilizado e provavelmente o algoritmo de agrupamento mais conhecido.

O objetivo do algoritmo K-Means é fornecer uma classificação automática sem a necessidade de supervisão, isto é, sem nenhuma pré-classificação existente. Por causa disso, esse algoritmo é considerado não supervisionado (MACQUEEN et al., 1967).

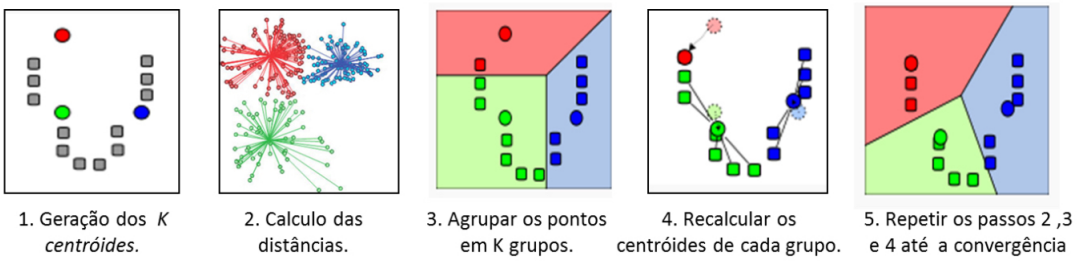
Exemplificando, o algoritmo funciona em cinco passos principais, conforme segue:

- Escolher os valores para os K centróides iniciais: neste passo, se escolhem arbitrariamente K pontos como os centros iniciais para cada grupo ou cluster;
- Geração da matriz de distâncias: neste passo, calcula-se a distância entre cada ponto e os centróides. Aqui, acontece a parte com maior processamento devido ao número de cálculos a se realizar, pois terão que ser calculadas $N.K$ distâncias;
- Agrupar os pontos de acordo com a sua distância: nessa parte, os pontos são agrupados em relação aos centróides, distribuindo-os ao grupo com quem possuem a menor distancia (maior similaridade). No caso que nenhum ponto seja incorporado a um grupo diferente de onde ele estava, o algoritmo termina.
- Calcular os novos centróides para cada grupo: após agrupar todos os pontos, os valores das coordenadas dos centróides são recalculados pela média de cada atributo

de todos os pontos que pertencem a esse cluster, procura-se encontrar uma partição melhor do que a gerada arbitrariamente.

- Repetir até a convergência: Repetir iterativamente desde o passo 2 até que os grupos convirjam. Isto é, refinando os grupos até que não existam mais alterações neles.

Figura 9 – Exemplo do funcionamento do algoritmo K-means para três grupos.



Fonte: (CÁRDENAS, 2015)

Na Figura 9, mostra-se um exemplo do funcionamento do algoritmo K- means para três grupos.

2.5 Materiais e Plataformas de Desenvolvimento

2.5.1 MATLAB

O Matlab foi escolhido como ferramenta para uma primeira implementação dos algoritmos. Esse *software* é conhecido mundialmente como uma excelente ferramenta para soluções de problemas matemáticos, científicos e tecnológicos, e possui comandos muito próximos da forma como se escrevem as expressões matemáticas. A linguagem é ideal para desenvolver rapidamente protótipos de novos programas.

O Matlab começou apenas como um *software* para operações matemáticas sobre matrizes, mas ao longo dos anos transformou-se em um sistema computacional flexível capaz de desenvolver essencialmente qualquer problema técnico. MATLAB possui uma vasta biblioteca de funções predefinidas, tais como: matemática elementar; funções especiais; matrizes elementares; matrizes especiais; decomposição e fatorização de matrizes; análise de dados; polinômios; solução de equações diferenciais; equações não-lineares e otimização; integração numérica; processamento de sinais entre outras.

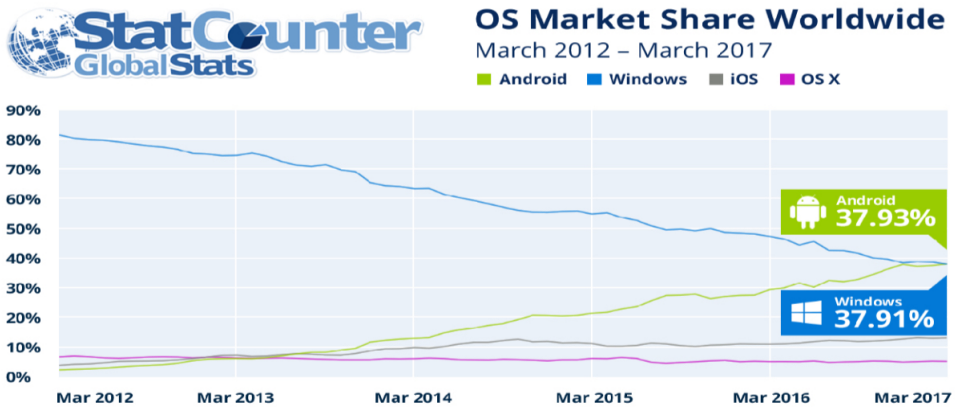
O livro de Theodoridis et al. (2010) foi usado como referência para a implementação, pois fornece funções do MATLAB para executar alguns dos algoritmos necessários (treinamento de Baum-Welch e K-means), possuindo também exemplos de como usar os scripts.

2.5.2 Aplicação Android

O aplicativo para Android usado como protótipo foi desenvolvido com o auxílio do framework Cocos2D-x. Esse framework foi idealizado originalmente para o desenvolvimento de jogos e aplicações gráficas, tendo como plataforma alvo os sistemas Android, IOS e Windows Phone. Devido ao suporte à linguagem "C", de mais baixo nível, e diversas otimizações realizadas, este framework é bastante utilizado em aplicações que têm requisitos de desempenho. Permite ainda a integração com bibliotecas de algoritmos nessa linguagem, o que foi de grande utilidade nessa monografia.

A plataforma/sistema Android é um sistema operacional baseado no Linux, projetado principalmente para dispositivos móveis como smartphones e tablets, sendo atualmente o sistema móvel mais utilizado no mundo em acessos à internet, segundo pesquisa realizada pela empresa StatCounter em março de 2017 (figura 10).

Figura 10 – Quota de mercado dos sistemas operacionais mais populares no mundo.



O código do sistema operacional Android é disponibilizado pela empresa Google sob licença de código aberto, o que permite o desenvolvimento de projetos sem custo de propriedade. Assim, optou-se pelo sistema Android para implementação do projeto. Em relação ao hardware, definiu-se o smartphone Zenfone 3 Zoom, comercializado pela empresa Asus, por utilizar o sistema Android e possuir diversos sensores, inclusive o sensor acelerômetro que será utilizado no projeto.

3 Proposta de Solução e Modelo

Nesse capítulo, será descrita a implementação do sistema de autenticação proposto. A arquitetura desenvolvida no software MATLAB e em aplicativo Android será apresentada.

A estrutura do presente capítulo está formatada de maneira a se abordar, na ordem em que se seguem, os seguintes itens: a apresentação geral do modelo proposto, o processo de amostragem do sensor acelerômetro, o filtro para pré-processamento, a quantização, o treinamento dos modelos ocultos de Markov e, por fim, o cálculo do score e o processo de decisão.

3.1 Apresentação Geral do Modelo Proposto

Devido à grande complexidade dos algoritmos de reconhecimento de padrões envolvidos, a implementação foi realizada primeiramente em software MATLAB, linguagem de alto nível focada no desenvolvimento científico, permitindo, assim, um desenvolvimento interativo.

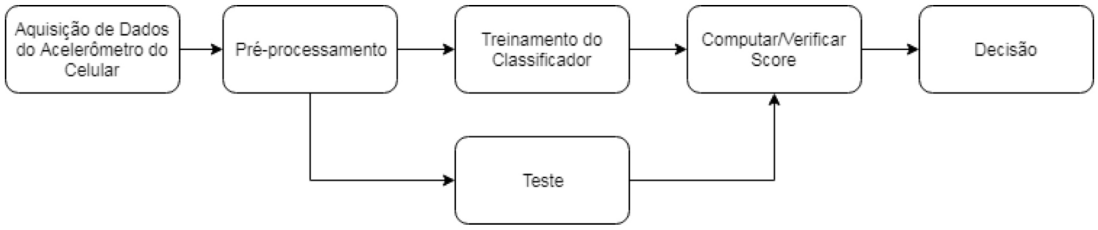
Assim, os algoritmos puderam ser rigorosamente testados e validados antes da tarefa mais complexa de implementar o aplicativo Android utilizado com protótipo.

O fluxo de execução do sistema de autenticação proposto pode ser dividido em cinco etapas principais:

- Aquisição de dados do acelerômetro: obtenção das amostras nos eixos x, y e z do sensor embutido no smartphone
- Pré-processamento: aplicação de filtro do tipo passa-alta para atenuação da constante de gravidade
- Quantização: utilização do algoritmo K-means para obtenção dos símbolos para entrada no modelo de Markov.
- Treinamento do classificador: os parâmetros do modelo oculto de Markov são calculados com base nas sequências de treinamento.
- Cálculo do score e Decisão: para uma nova sequência de amostras de um gesto um score é calculado e o classificador deve tomar uma decisão.

A Fig. 11 mostra a sequência dessas etapas. Cada estágio será explicado nas seções a seguir.

Figura 11 – Etapas do sistema de autenticação proposto.



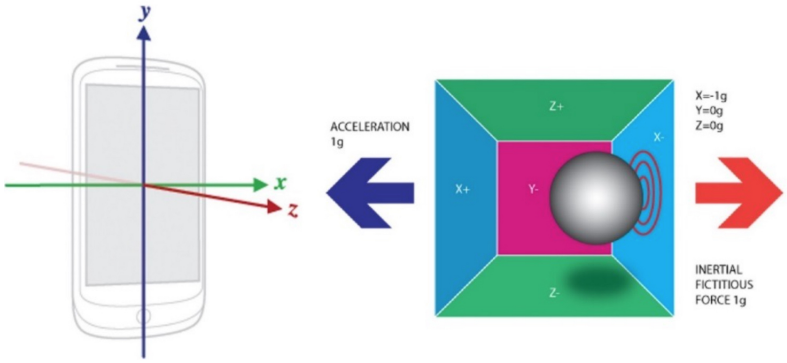
3.2 Descrição das Etapas do Modelo

3.2.1 Aquisição de dados do acelerômetro

O acelerômetro é um tipo de sensor de movimento que mede a força de aceleração em (m/s^2) ao qual é aplicada a um dispositivo.

Esse sensor pode ser pensado como uma caixa em forma de um cubo com uma bola em seu interior (Fig. 12). Quando uma força é aplicada às faces do cubo, a bola no interior irá flutuar na direção onde a pressão está localizada.

Figura 12 – Modelo de acelerômetro triaxial.



O sensor acelerômetro fornece valores em termos da gravidade g, o que significa que um valor de 1g corresponde a uma aceleração em torno de $(9,8m/s^2)$. Por exemplo, se o smartphone estiver ocioso na parte superior de uma mesa, os valores a seguir serão lidos do sensor, idealmente: x = 0g, y = 0g e z = + 1g.

Para capturar amostras do acelerômetro de diferentes gestos para a implementação em MATLAB, um simples aplicativo Android foi criado usando a linguagem de programação Java. Diferentes gestos foram executados várias vezes, e as sequências de amostras obtidas foram gravadas em arquivos, cada uma com um identificador do gesto e um número, a fim de se diferenciar as sequências da mesma classe de gestos. O período de amostragem do acelerómetro foi de 150 ms.

Um exemplo de sequência de amostras do sensor é mostrada na Fig. 13. Nessa figura, a linhas 1, 2 e 3 representam, respectivamente, os valores obtidos do acelerômetro

nos eixox X, Y e Z. As colunas, por sua vez, mostram diferentes amostras obtidas de forma sequencial na execução de um gesto.

Figura 13 – Amostras de um gesto do sensor acelerômetro.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.5625	0.8027	1.1260	0.8984	0.4785	0.2471	-0.0713	-0.1250	-0.2500	-0.2246	-0.2227	-0.1465	0.0264	0.4023	0.8701
2	0.1289	0.1123	0.0674	0.0117	0.0703	0.0225	-0.1416	0.0078	-0.0352	-0.0107	-0.0166	-0.0186	0.0029	-0.0029	-0.0859
3	-0.7754	-0.8887	-0.8721	-0.8672	-1.1309	-1.0518	-1.0205	-1.1113	-0.9150	-1.1113	-0.8896	-0.8877	-0.7920	-0.8223	-0.9756

As sequências de amostras obtidas, então, devem ser pré-processadas e quantizadas, conforme é explicado a seguir.

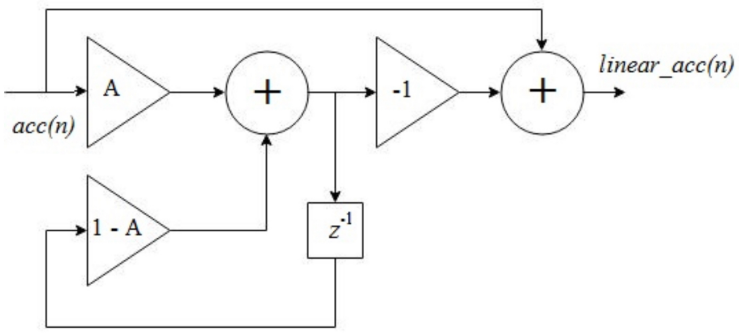
3.2.2 Pré-processamento

Antes de prosseguir para o estágio de quantização, foi necessário filtrar os dados do acelerômetro de entrada. Isso é preciso porque as amostras de aceleração incluem a constante de gravidade, que adiciona um deslocamento às medidas.

Percebeu-se que esse fato gera um problema na etapa de quantização e resultados de reconhecimento inconsistentes, que foram verificados durante os testes. Portanto, a solução encontrada foi processar os dados recebidos usando um filtro passa-alta, de forma a diminuir os efeitos da constante de gravidade.

Um filtro passa-alta é aquele que atenua as frequências mais baixas e os sinais de constantes. O modelo do filtro utilizado foi baseado naquele encontrado na documentação do sistema operacional Android (GOOGLE, 2018). A figura 14 mostra o modelo do filtro. Neste, a constante A, mostrada no filtro, pode ser escolhida conforme o nível de atenuação desejado. A própria documentação recomenda o valor de 0.8, que foi utilizado neste trabalho.

Figura 14 – Filtro passa-altas usado para atenuar os efeitos da constante gravitacional nas medidas de aceleração.



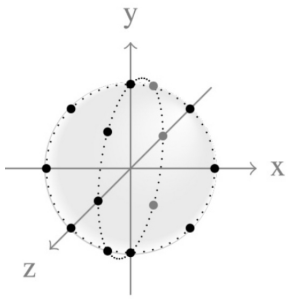
3.2.3 Quantização

Antes de treinar os Modelos Ocultos de Markov, houve a necessidade de reduzir o número de dimensões e discretizar as amostras do acelerômetro. Portanto, a solução foi quantizar as amostras pré-processadas, isto é, atribuir a cada amostra um símbolo. Isso foi necessário porque um HMM discreto foi utilizado para representar um gesto.

A quantização foi realizada para cada sequência separadamente. O algoritmo usado neste estágio foi o K-means, que classifica cada amostra do acelerômetro triaxial em um dos clusters definidos previamente.

O conjunto de símbolos ou clusters disponíveis é freqüentemente chamado de codebook (livro de códigos). A localização inicial de cada cluster também deve ser cuidadosamente especificada e depende do problema que se está tentando resolver. Os parâmetros utilizados no projeto foram baseados nos apresentados no trabalho de Schlömer et al. (2008), no qual quatorze clusters são definidos no espaço tridimensional em torno de pontos em uma esfera, como visto na figura 15. O raio da esfera é calculado dinamicamente para cada sequência, como sendo o raio médio de todas as amostras.

Figura 15 – As localizações iniciais dos clusters para K = 14 (SCHLöMER et al., 2008).



Na saída do estágio de quantização, a cada amostra de acelerômetro pré-processada de uma sequência é atribuído um número de zero a treze, que representa um cluster. A figura 16 mostra o exemplo de uma sequência antes e depois da quantização.

Figura 16 – Exemplo de sequência de amostras de um gesto sendo quantizada.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.5625	0.8027	1.1260	0.8984	0.4785	0.2471	-0.0713	-0.1250	-0.2500	-0.2246	-0.2227	-0.1465	0.0264	0.4023	0.8701
2	0.1289	0.1123	0.0674	0.0117	0.0703	0.0225	-0.1416	0.0078	-0.0352	-0.0107	-0.0166	-0.0186	0.0029	-0.0029	-0.0859
3	-0.7754	-0.8887	-0.8721	-0.8672	-1.1309	-1.0518	-1.0205	-1.1113	-0.9150	-1.1113	-0.8896	-0.8877	-0.7920	-0.8223	-0.9756

⇩ K-means

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	8	8	8	8	8	7	7	7	7	7	7	7	7	8	8

Neste exemplo, as variações do valores apresentados a cada amotra indicam que o smartphone está se movendo em linhas retas durante o gesto, em duas direções diferentes. As colunas representam diferentes amostras, e as linhas 1, 2 e 3 ao topo representam as acelerações nos eixo X, Y e Z, que são então quantizadas.

3.2.4 Treinamento do Modelo Oculto de Markov

De acordo com Theodoridis et al. (2010), um Modelo Oculto de Markov(HMM) é um tipo de modelagem estocástica apropriada para sequências estocásticas não estacionárias cujas propriedades estatísticas sofrem transições aleatórias entre um conjunto de N diferentes processos estacionários. N pode ser pensado como o número de fontes, e cada fonte pode ser associada a um estado.

Em HMMs Discretos, os parâmetros são probabilidades brutas, enquanto que em HMMs Contínuos, os parâmetros são associados a funções de densidade de probabilidade (THEODORIDIS et al., 2010). Um HMM Discreto foi usado e pode ser descrito pelo seguinte conjunto de parâmetros:

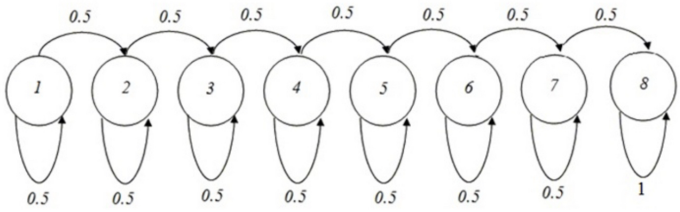
- O número de estados N;
- O número de símbolos K;
- A Matriz de probabilidades de observação de símbolos, B, de tamanho $N \times K$, a qual especifica a probabilidade de um símbolo específico ser emitido em cada estado, incluindo todas as possíveis combinações.
- A Matriz de transição de estados, A, de tamanho $N \times N$, em que um elemento a_{ij} especifica a probabilidade do sistema pular do estado atual i para outro estado j.
- O vetor de distribuição inicial de estados, de tamanho N, que contém a probabilidade de cada estado ser o estado inicial.

Um HMM Discreto foi usado para representar um modelo de gesto, e seus parâmetros são calculados com base em um modelo inicial e algumas sequências de observação. O uso de várias seqüências para treinamento é uma prática comum, onde diferentes instâncias de realização do processo estocástico são exploradas em uma lógica de média (RABINER, 1989).

Dado um conjunto de sequências de observação geradas pelo usuário, os parâmetros desconhecidos do HMM são estimados usando uma estimação de parâmetro de máxima verossimilhança, também chamada de algoritmo de treinamento de Baum-Welch, para determinar os melhores parâmetros possíveis para as seqüências de observação fornecidas.

O modelo inicial adotado baseou-se naquele encontrado no trabalho de Schlömer et al. (2008), que é um modelo da esquerda para a direita usando 8 estados, como mostrado na figura 17. Um modelo da esquerda para a direita é aquele em que não há transição de estado de um estado de índice mais alto para um mais baixo, isto é, sua matriz de transição de estado é triangular superior (RABINER, 1989). Alguns outros modelos foram testado

Figura 17 – O modelo inicial usado na etapa de treinamento.



O modelo inicial usado é exemplificado na figura 17. Sua matriz de transição de estados correspondente é mostrada na equação 3.1.

$$A = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

No vetor de probabilidade de estado inicial π, a probabilidade do primeiro estado é definida como um e os outros como zero, já que o sistema deve sempre iniciar a partir do primeiro estado, como mostrado na equação 3.2.

$$\pi = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.2}$$

3.2.5 Cálculo do score e decisão

A última etapa do sistema de autenticação baseado em gestos proposto é o cálculo do score e a decisão.

A autenticação tem como objetivo principal verificar se o usuário é quem diz ser. Para tal, no caso deste projeto, ao realizar uma tentativa de autenticação, o gesto executado deve ser comparado com o modelo previamente calculado. O classificador deve

então, a partir de uma métrica, decidir se o gesto corresponde ao modelo armazenado ou não.

Dada uma sequência de amostras de um gesto e um HMM treinado, é possível calcular a probabilidade de que a sequência tenha sido produzida pelo modelo (RABINER, 1989). Essa probabilidade resultante é dimensionada e usada como pontuação. Se a pontuação for maior que um certo limite, isso é considerado uma correspondência.

O limiar, ou *threshold*, deve ser escolhido de forma que se reduza a quantidade de falsos positivos e falsos negativos. Como esses são inversamente proporcionais, busca-se, como já detalhado, a menor *equal error rate* (EER), que corresponde ao valor de limiar em que a quantidade de falsos positivos é igual à de falsos negativos. A EER é um importante parâmetro para validação da biometria proposta. Este e outros parâmetros obtidos são apresentados no próximo capítulo.

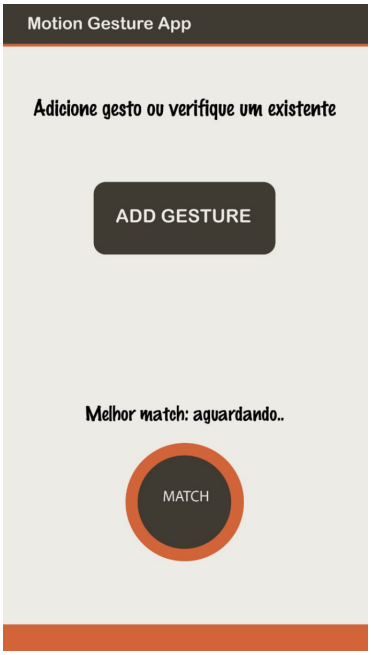
4 Resultados

4.1 Protótipo para Android

O aplicativo desenvolvido como protótipo possibilita o registro de gestos variados e, após o cadastro de alguns, realizar um gesto e verificar se o resultado corresponde ao gesto executado. A seguir se detalha a operação do aplicativo.

A figura 18 mostra a tela inicial do aplicativo desenvolvido. Ao se pressionar o botão para adicionar um gesto (Add Gesture), ao topo, uma nova tela é aberta. Essa tela de cadastro, figura 19, possibilita adicionar um gesto qualquer. Para tal, deve-se pressionar o botão iniciar e, mantendo-o pressionado, executar o movimento desejado. O mesmo deve ser repetido algumas vezes para treinar o modelo. O número de 6 repetições foi definido, pois se demonstrou razoável e apresentou resultados satisfatórios, não sendo tão inconveniente para o usuário.

Figura 18 – Tela inicial do aplicativo de teste desenvolvido para Android.



Alguns gestos diferentes devem ser cadastrados, da forma como explicado. Após ter-se um número razoável de gestos cadastrados, é possível, na tela iniciar, executar um dos gestos e verificar se o gesto correto foi identificado.

A primeira versão da implementação não incluiu o estágio de pré-processamento, ou seja, o filtro passa-altas. Em razão disso, os resultados obtidos nos primeiros testes foram inconsistentes. A solução se mostrava sensível a qualquer pequena variação no gesto

que estava sendo executado e a correspondência correta não era encontrada na maioria das vezes.

Figura 19 – Tela de cadastro de gestos do aplicativo de teste desenvolvido para Android.

Após o filtro passa-altas ser implementado, os resultados se tornaram bem mais consistentes. Isso se deve ao fato de esse filtro atenuar os efeitos da aceleração da gravidade, como já explicado.

Testes foram realizados para encontrar a melhor combinação dos parâmetros de algoritmos descritos anteriormente: o número de estados no HMM inicial (8), número de símbolos no livro de códigos (14), período de amostragem do acelerômetro (150 ms).

4.2 Validação em MATLAB

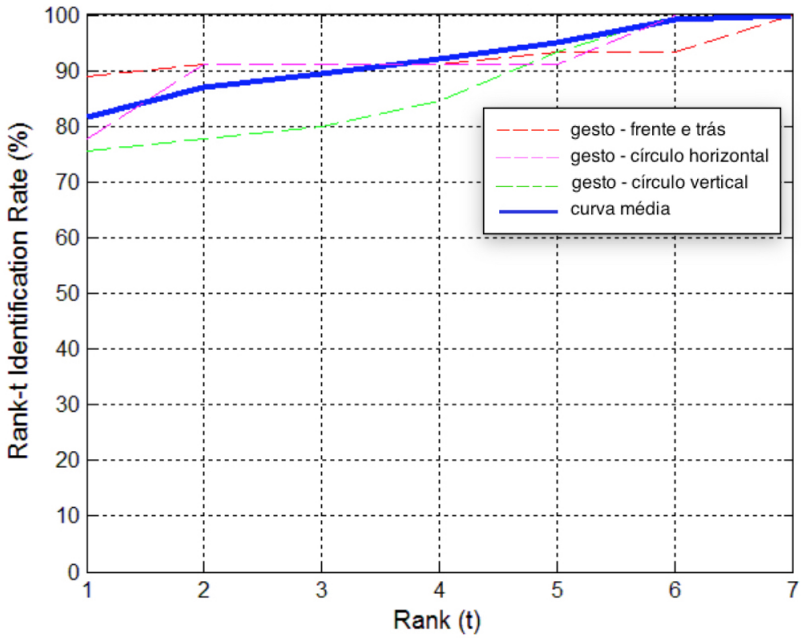
A validação foi realizada de forma conjunta ao processo de desenvolvimento. Isso facilitou bastante a localização e a correção de erros.

Para avaliar o sistema de autenticação, sete classes diferentes de gestos foram usadas, incluindo círculos horizontais e verticais no sentido horário e anti-horário, para frente e para trás, para cima e para baixo, e alguns padrões mais complexos, como uma assinatura no ar.

Cada classe de gesto foi realizada 70 vezes. Um modelo oculto de Markov foi treinado para cada classe usando 20 gestos/seqüências. As 50 sequências remanescentes de cada classe foram usadas para construir a matriz de pontuação de similaridade, com um total de 2450 pontos. A Figura 20 mostra a curva característica cumulativa de correspondência (CMC) para este conjunto de dados. A taxa de rank 1 foi de 81%. Como explicado

anteriormente, a curva CMC sintetiza o desempenho da biometria em relação ao problema de identificação. Assim, entende-se que, em 81% dos casos, a classe de gesto correta foi identificada.

Figura 20 – Curva CMC (Cumulative match characteristic). As linhas tracejadas são curvas CMC ao se considerar alguns gestos isoladamente. A linha sólida azul é a curva CMC final.



Diversas curvas Receiver Operating Characteristic (ROC) são mostradas na figura 21 para diferentes quantidades de seqüências de treinamento. Para 30 sequências de treino, o sistema atinge uma taxa de aceitação genuína de 90% a uma taxa de aceitação falsa de cerca de 40%.

As distribuições genuína e de impostor são mostradas na Fig. 22. Como detalhado no referencial teórico, idealmente essas distribuições devem ser separadas o máximo possível. Pode ser visto, porém, que as distribuições obtidas têm uma clara e representativa sobreposição em um score de cerca de -25.

A Equal Error rate (EER) foi calculada e chegou-se a um resultado de 20% em um score de -26. Esse valor representa uma taxa de aceitação genuína de 80% a uma taxa de aceitação falsa de 20%.

Os testes realizados no aplicativo protótipo desenvolvido foram compatíveis com os resultados obtidos nas simulações em MATLAB. Desse modo, a solução se mostrou adequada na aplicação da autenticação biométrica para dispositivos móveis usando a biometria comportamental por gestos manuais.

Figura 21 – Curvas ROC (Receiver Operating Characteristic) para 10, 15, 20 e 30 seqüências de treinamento.

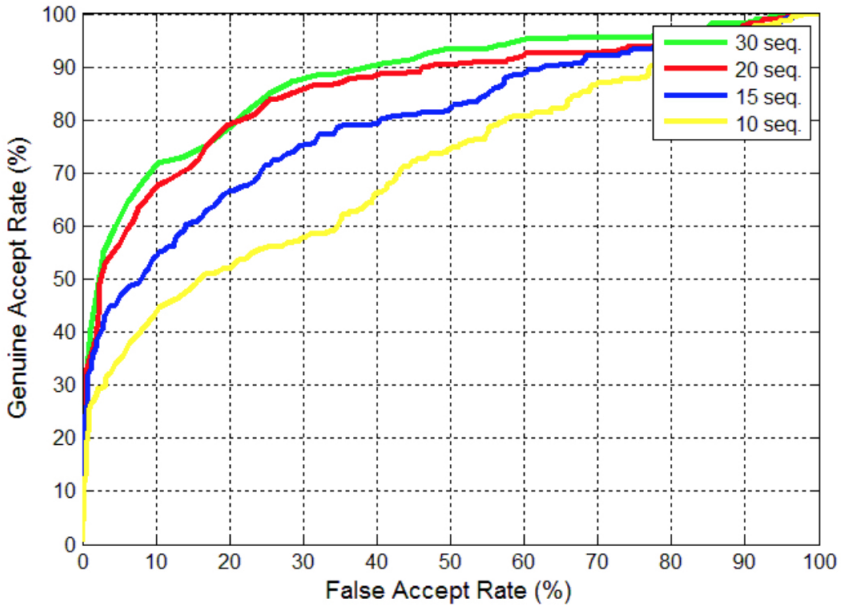
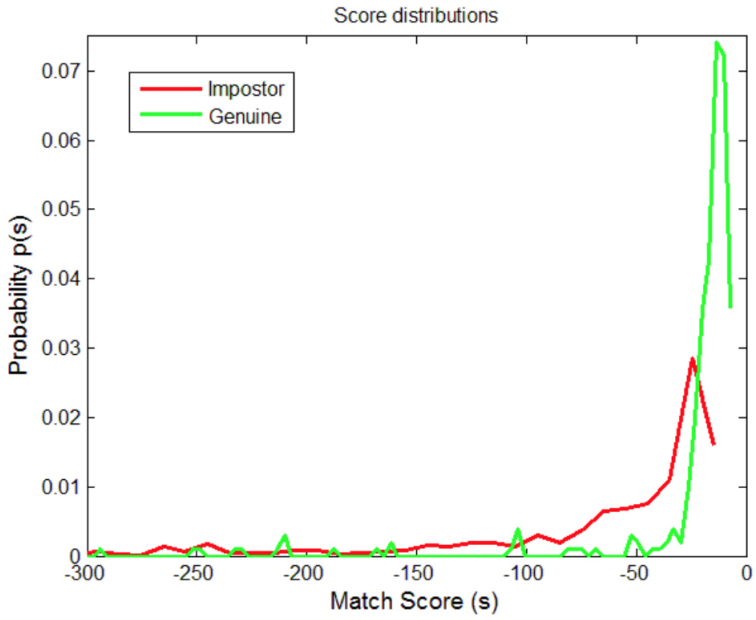


Figura 22 – Distribuições genuína e impostora para 30 seqüências de treinamento.



5 Conclusão

Este trabalho visou investigar a aplicação de biometria comportamental baseada em gestos manuais, como alternativa aos métodos tradicionais de autenticação.

Assim, foi proposto e implementado um sistema biométrico de autenticação para dispositivos móveis baseado em reconhecimento de gestos manuais a partir do uso de sensor acelerômetro.

Descreveu-se, também, o processo de desenvolvimento, bem como as ferramentas necessárias para implementação do projeto.

O sistema proposto obteve uma taxa de identificação razoável, com rank-1 de 81% para sete gestos variados, incluindo círculos horizontais e verticais no sentido horário e anti-horário, para frente e para trás, para cima e para baixo, e alguns padrões mais complexos, como uma assinatura no ar. A execução dos gestos foi realizada em ambiente sem influência de aceleração externa.

Ao se analisar a curva ROC e os gráficos de distribuição, porém, observa-se altas taxas de falsos positivos, o que dificulta o uso da biometria proposta como uma solução de autenticação independente.

Percebeu-se que, ao se variar alguns parâmetros do sistema, por exemplo, o número de estados do modelo de Markov ou o limite de saltos, produz-se resultados consideravelmente diferentes. Além disso, ao incluir gestos mais complexos no conjunto de dados, também se diminui significativamente o desempenho do sistema.

Desta forma, conclui-se que a solução proposta poderia ser usada como uma camada adicional de segurança, junto a um token ou senha ou a outra biometria. Um possível projeto futuro poderia envolver o uso da solução biométrica aplicada nesta monografia em conjunto a outras biometrias já estabelecidas tradicionalmente, ou outras formas de autenticação. A influência de aceleração externa durante a execução dos gestos também poderia ser investigada. Outro possível projeto futuro poderia visar a aplicação de vários sensores embutidos (acelerômetro, giroscópio, bússola, etc) e técnicas de fusão de sensores, no qual se acredita que uma melhoria considerável de desempenho poderia ser alcançada.

Referências

BHAGAVATULA, C. et al. Biometric authentication on iphone and android: Usability, perceptions, and influences on adoption. 2015.

CÁRDENAS, E. J. E. Desenvolvimento de uma abordagem para o reconhecimento de gestos manuais dinâmicos e estáticos. 2015.

CLARKE, N. L.; FURNELL, S. M. Authentication of users on mobile telephones–a survey of attitudes and practices. *Computers & Security*, Elsevier, v. 24, n. 7, p. 519–527, 2005.

COSTA, L. R.; OBELHEIRO, R. R.; FRAGA, J. S. Introdução á biometria. *Livro texto dos Minicursos do VI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg2006). SBC: Porto Alegre*, v. 1, p. 103–151, 2006.

DELAC, K.; GRGIC, M. A survey of biometric recognition methods. In: *46th International Symposium Electronics in Marine*. [S.l.: s.n.], 2004. v. 46, p. 16–18.

DOMINGOS, N. d. S. *Biometrias comportamentais em dispositivos móveis*. Tese (Doutorado), 2014.

ESPINDOLA, L. d. S. Um estudo sobre modelos ocultos de markov hmm-hidden markov model. *Pontifícia Universidade Católica do Rio Grande do Sul, Faculdade de Informática, Porto Alegre*, 2009.

FARELLA, E. et al. Gesture signature for ambient intelligence applications: A feasibility study. In: FISHKIN, K. et al. (Ed.). *Pervasive Computing*. Springer Berlin Heidelberg, 2006. (Lecture Notes in Computer Science, v. 3968), p. 288–304. ISBN 978-3-540-33894-9. Disponível em: <http://dx.doi.org/10.1007/11748625_18>.

GOOGLE. *Android Motion Sensors*. 2018. Disponível em: <https://developer.android.com/guide/topics/sensors/sensors_motion>.

GRIGOLETTI, P. S. Cadeias de markov. *Recuperado em*, v. 19, n. 10, p. 2014, 2011.

HANLEY, J. A.; MCNEIL, B. J. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, v. 143, n. 1, p. 29–36, 1982.

JAIN, A. K.; ROSS, A.; PRABHAKAR, S. An introduction to biometric recognition. *IEEE Transactions on circuits and systems for video technology*, IEEE, v. 14, n. 1, p. 4–20, 2004.

JORGENSEN, Z.; YU, T. On mouse dynamics as a behavioral biometric for authentication. In: *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. New York, NY, USA: ACM, 2011. (ASIACCS '11), p. 476–482. ISBN 978-1-4503-0564-8. Disponível em: <http://doi.acm.org/10.1145/1966913.1966983>.

KELA, J. et al. Accelerometer-based gesture control for a design environment. *Personal Ubiquitous Comput.*, Springer-Verlag, London, UK, UK, v. 10, n. 5, p. 285–299, jul. 2006. ISSN 1617-4909. Disponível em: <http://dx.doi.org/10.1007/s00779-005-0033-8>.

LIU, J. et al. uwave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, Elsevier, v. 5, n. 6, p. 657–675, 2009.

LIU, S.; SILVERMAN, M. A practical guide to biometric security technology. *IT Professional*, IEEE, v. 3, n. 1, p. 27–32, 2001.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. [S.l.], 1967. v. 1, n. 14, p. 281–297.

MAGALHÃES, P. S. T.; SANTOS, H. D. d. Biometria e autenticação. Associação Portuguesa de Sistemas de Informação (APSI), 2003.

MäNTYJäRVI, J. et al. Enabling fast and effortless customisation in accelerometer based gesture interaction. In: *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia*. New York, NY, USA: ACM, 2004. (MUM '04), p. 25–31. ISBN 1-58113-981-0. Disponível em: <http://doi.acm.org/10.1145/1052380.1052385>.

MAYRON, L. Biometric authentication on mobile devices. *Security and Privacy, IEEE*, v. 13, n. 3, p. 70–73, May 2015. ISSN 1540-7993.

OLIVEIRA, L. E. S. de; MORITA, M. E. Introduçao aos modelos escondidos de markov (hmm). 2000.

PATEL, S. N.; PIERCE, J. S.; ABOWD, G. D. A gesture-based authentication scheme for untrusted public terminals. In: *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA: ACM, 2004. (UIST '04), p. 157–160. ISBN 1-58113-957-8. Disponível em: <http://doi.acm.org/10.1145/1029632.1029658>.

PEACOCK, A.; KE, X.; WILKERSON, M. Typing patterns: a key to user identification. *Security and Privacy, IEEE*, v. 2, n. 5, p. 40–47, Sept 2004. ISSN 1540-7993.

PRABHAKAR, S.; PANKANTI, S.; JAIN, A. Biometric recognition: security and privacy concerns. *Security and Privacy, IEEE*, v. 1, n. 2, p. 33–42, Mar 2003. ISSN 1540-7993.

PYLVÄNÄINEN, T. Accelerometer based gesture recognition using continuous hmms. In: SPRINGER. *Iberian Conference on Pattern Recognition and Image Analysis*. [S.l.], 2005. p. 639–646.

RABINER, L. R. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, Ieee, v. 77, n. 2, p. 257–286, 1989.

RAUTARAY, S. S.; AGRAWAL, A. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, Springer, v. 43, n. 1, p. 1–54, 2015.

SCHLöMER, T. et al. Gesture recognition with a wii controller. In: *Proceedings of the 2Nd International Conference on Tangible and Embedded Interaction*. New York, NY, USA: ACM, 2008. (TEI '08), p. 11–14. ISBN 978-1-60558-004-3. Disponível em: <http://doi.acm.org/10.1145/1347390.1347395>.

SILVA, A. P. Modelos ocultos de markov. 2015.

THEODORIDIS, S. et al. *Introduction to pattern recognition: a matlab approach*. [S.l.]: Academic Press, 2010.

TREWIN, S. et al. Biometric authentication on a mobile device: A study of user effort, error and task disruption. In: *Proceedings of the 28th Annual Computer Security Applications Conference*. New York, NY, USA: ACM, 2012. (ACSAC '12), p. 159–168. ISBN 978-1-4503-1312-4. Disponível em: <http://doi.acm.org/10.1145/2420950.2420976>.

WANG, J.-S.; CHUANG, F.-C. An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition. *Industrial Electronics, IEEE Transactions on*, v. 59, n. 7, p. 2998–3007, July 2012. ISSN 0278-0046.

WAYMAN, J. et al. An introduction to biometric authentication systems. In: *Biometric Systems*. [S.l.]: Springer, 2005. p. 1–20.

WU, J. et al. Gesture recognition with a 3-d accelerometer. In: *Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing*. Berlin, Heidelberg: Springer-Verlag, 2009. (UIC '09), p. 25–38. ISBN 978-3-642-02829-8. Disponível em: <http://dx.doi.org/10.1007/978-3-642-02830-4_4>.

XU, R.; ZHOU, S.; LI, W. J. Mems accelerometer based nonspecific-user hand gesture recognition. *Sensors Journal, IEEE*, IEEE, v. 12, n. 5, p. 1166–1173, 2012.

Anexos

ANEXO A – Código MATLAB

Listing A.1 – Rotina principal de cálculo de scores e matriz de similaridade

```
1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  % Monografia
3  % Matheus Bichara de Assumpcao
4  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6  % Clean
7  clear; clc; close all;
8
9  %%%%%%%%%%%%%%%%%%%% CONSTANTES %%%%%%%%%%%%%%%%%%%%
10
11 numState = 8; % # de estados Hidden Markov Model
12
13 numGestures = 7; % Numero de gestos(classes)
14
15 n_training = 20; % Numero de sequencias usadas para treinamento
16 n_matching = 50; % Numero de sequencias utilizadas para computar os scores
17
18 % Pontos na curva CMC
19 CMC_points = numGestures;
20 % Pontos na curva ROC
21 ROC_points = 2000;
22 % Quantidade de bins no grafico de distribuicao
23 Nbins = 100;
24
25 %%%%%%%%%%%%%%%%%%%%%% INIcio %%%%%%%%%%%%%%%%%%%%%%
26
27 dataAmount = n_training + n_matching; %Total de sequencias
28
29 % Obtendo sequencia de amostras de acelerometro para cada classe de gesto (
       ver parse_acc_data.m)
30 [gesture{1}.accData] = parse_acc_data(dataAmount, \ 'upDown/upDown\');
31 [gesture{2}.accData] = parse_acc_data(dataAmount, \ 'forBack/forBack\');
32 [gesture{3}.accData] = parse_acc_data(dataAmount, \ 'circleClock/
       circleClock\');
33 [gesture{4}.accData] = parse_acc_data(dataAmount, \ 'circleAnti/circleAnti\
       ');
34 [gesture{5}.accData] = parse_acc_data(dataAmount, \ 'squareClock/
       squareClock\');
35 [gesture{6}.accData] = parse_acc_data(dataAmount, \ 'triAnti/triAnti\');
36 [gesture{7}.accData] = parse_acc_data(dataAmount, \ 'sigM/sigM\');
37
```

```
38 % Quantizando e treinando um Hidden Markov Model para cada gesto usando \'
       n_training\' sequencias
39 for g = 1:numGestures
40     [gesture{g}.pi, gesture{g}.A, gesture{g}.B, ~] = trainHMM(gesture{g}.
           accData(1:n_training), alpha, numState);
41 end
42
43 % Quantizando as sequencias restantes
44 for g = 1:numGestures
45     [gesture{g}.Ob] = quantize(gesture{g}.accData(n_training + 1:dataAmount
           ), alpha);
46 end
47
48 % Calculando as matrizes de score de similaridade
49 for seq = 1:n_matching
50     for g = 1:numGestures
51         for k = 1:numGestures
52             scores{g}(seq, k) = BWDoHMMsc(gesture{k}.pi, gesture{k}.A,
                   gesture{k}.B, gesture{g}.Ob{seq});
53         end
54     end
55 end
56
57 % Separando scores genuinos de impostores
58 impostor_scores = [];
59 genuine_scores = [];
60 for g = 1:numGestures
61
62     scores{g}(isnan(scores{g})) = -1000; % Removendo respostas com erro
63     scores{g}(isinf(scores{g})) = -1000;
64
65     genuine_scores{g} = scores{g}(:, g);
66
67     impostor_scores{g} = scores{g};
68     impostor_scores{g}(:, g) = []; % Removendo coluna de scores genuinos
69     impostor_scores{g} = impostor_scores{g}(:);
70
71 end
72
73 %Calculando as taxas rank-t usando dados de gesto especifico
74 for g = 1:numGestures
75     [rank_t_rate{g}] = calcRankT(scores{g}, genuine_scores{g}, n_matching);
76 end
77
78 %Achando a rank-t geral
79 rank = 0;
80 for g = 1:numGestures
```

```
81      rank = rank + rank_t_rate{g};
82  end
83  rank = rank / 7;
84
85  %plotando a curva CMC
86  temp = sprintf(\ 'CMC curve\');
87  figure(\ 'name\', temp);
88  plot(rank, \ 'b\', \ 'LineWidth\', 3);
89
90  grid on
91
92  title(temp)
93  ylabel(\ '\\fontsize{12}Rank-t Identification Rate (%)\');
94  xlabel(\ '\\fontsize{12}Rank (t)\');
95  axis([1, CMC_points, 0, 100]);
96
97  %Concatenando todos os dados genuinos e impostor
98  gen = [];
99  imp = [];
100 for g = 1:numGestures
101     gen = [gen; genuine_scores{g}];
102     imp = [imp; impostor_scores{g}];
103 end
104 gen = gen(gen ~= -1000);
105 imp = imp(imp ~= -1000);
106
107 % Calculando far, frr e equal error rate
108 min_thr = min(min([imp; gen]));
109 max_thr = max(max([imp; gen]));
110 [far, frr, thr_range, eer] = calcFarFrr(gen, imp, min_thr, max_thr,
        ROC_points);
111
112 % GAR da FRR
113 gar = ones(1, ROC_points) * 100 - frr;
114
115 % Plotando Curva ROC
116 temp = sprintf(\ 'ROC curve\');
117 figure(\ 'name\', temp);
118 plot(far, gar, \ 'b\', \ 'LineWidth\', 3);
119 grid on
120 title(temp)
121 ylabel(\ '\\fontsize{12}Genuine Accept Rate (%)\');
122 xlabel(\ '\\fontsize{12}False Accept Rate (%)\');
123 axis([0, 100, 0, 100]);
124
125 % Calculando as Distribuicoes
126 [dist] = calcDists(gen, imp, Nbins);
```

```
127
128 %Plotando as distribuicoes
129 temp = sprintf(\ 'Score distributions\');
130 figure(\ 'name\', temp)
131 bar(dist.imp_centers, dist.imp_count_norm, \ 'r\', \ 'LineWidth\', 1);
132 hold on
133 bar(dist.gen_centers, dist.gen_count_norm, \ 'g\', \ 'LineWidth\', 1);
134 hold off
135 %alpha(.6);
136 title(temp)
137 ylabel(\ '\\fontsize{12}Probability p(s)\');
138 xlabel(\ '\\fontsize{12}Match Score (s)\');
139 legend(\ 'Impostor\', \ 'Genuine\')
140 axis([- 400 0 0 inf])
141
142 % end
```

Listing A.2 – Rotina de leitura dos dados do acelerômetro

```
1  function [ Acc ] = parse_acc_data( numOfSequencesToCalculate, path_begin )
2  %PARSE_ACC_DATA Ler sequencia de amostras do arquivo
3
4
5  formatSpec = 'x:%f y:%f z:%f\n';
6  sizeCount = [3 Inf];
7  for i=1:numOfSequencesToCalculate
8      file_path = strcat(path_begin,num2str(i),'.txt');
9      fp = fopen(file_path,'r');
10
11     accData = fscanf(fp,formatSpec,sizeCount);
12     accData = accData';
13
14     Acc{i}(:,1) = accData(:,1);
15     Acc{i}(:,2) = accData(:,2);
16     Acc{i}(:,3) = accData(:,3);
17     fclose(fp);
18
19 end
20
21 end
```

Listing A.3 – Rotina para quantização

```
1  function [bel] = quantize(Acc, alpha)
2  %QUANTIZACAO
3
4  for i = 1:length(Acc)
5      % Removendo Primeira linha
6      Acc{i}(1, :) = [];
```

```
7  end
8
9  % QUANTIZACAO
10 for i = 1:length(Acc)
11
12     % Raio da esfera
13     for j = 1:length(Acc{i})
14         tempVec(j) = sqrt(Acc{i}(j, 1) ^ 2 + Acc{i}(j, 2) ^ 2 + Acc{i}(j,
               3) ^ 2);
15     end
16     radius = (max(tempVec) + min(tempVec)) / 2;
17
18     % Setando posicao inicial dos clusters
19     theta_init(:, 1) = [radius 0 0];
20     theta_init(:, 2) = [cos(pi / 4) * radius 0 sin(pi / 4) * radius];
21     theta_init(:, 3) = [0 0 radius];
22     theta_init(:, 4) = [cos(pi * 3 / 4) * radius 0 sin(pi * 3 / 4) * radius
           ];
23     theta_init(:, 5) = [- radius 0 0];
24     theta_init(:, 6) = [cos(pi * 5 / 4) * radius 0 sin(pi * 5 / 4) * radius
           ];
25     theta_init(:, 7) = [0 0 - radius];
26     theta_init(:, 8) = [cos(pi * 7 / 4) * radius 0 sin(pi * 7 / 4) * radius
           ];
27     theta_init(:, 9) = [0 radius 0];
28     theta_init(:, 10) = [0 cos(pi / 4) * radius sin(pi / 4) * radius];
29     theta_init(:, 11) = [0 cos(pi * 3 / 4) * radius sin(pi * 3 / 4) *
           radius];
30
31     theta_init(:, 12) = [0 - radius 0];
32     theta_init(:, 13) = [0 cos(pi * 5 / 4) * radius sin(pi * 5 / 4) *
           radius];
33     theta_init(:, 14) = [0 cos(pi * 7 / 4) * radius sin(pi * 7 / 4) *
           radius];
34
35     Acc{i} = Acc{i} \ ';
36
37 ␣␣␣␣%Rodando␣o␣algoritmo␣K-means
38 ␣␣␣␣[~,␣bel{i},␣~]␣=␣k_means(Acc{i},␣theta_init);
39
40 end
41
42 end
```

Listing A.4 – Rotina de treinamento do modelo de Markov

```
1  function [piTrained_1, ATrained_1, BTrained_1, bel ] = trainHMM( Acc, alpha
       , NumStates)
```

```
2  %trainHMM
3
4  m = 14; % Numero de clusters
5  maxEpoch=500; % Numero maximo de iteracoes
6
7
8  for i=1:length(Acc)
9      % Remove primeira linha
10     Acc{i}(1,:) = [];
11 end
12
13 % Quantizacao - Kmeans
14 for i=1:length(Acc)
15
16     % Achando raio da esfera
17     for j=1:length(Acc{i})
18         tempVec(j) = sqrt( Acc{i}(j,1)^2 + Acc{i}(j,2)^2 + Acc{i}(j,3)^2
               );
19     end
20     radius = (max(tempVec) + min(tempVec))/2;
21
22     % Setando as posicoes iniciais dos clusters
23     theta_init(:, 1) = [radius   0   0 ];
24     theta_init(:, 2) = [cos(pi / 4)*radius   0   sin(pi / 4)*radius ];
25     theta_init(:, 3) = [0   0   radius ];
26     theta_init(:, 4) = [cos(pi * 3 / 4)*radius   0   sin(pi * 3 / 4)*radius
           ];
27     theta_init(:, 5) = [-radius   0   0 ];
28     theta_init(:, 6) = [cos(pi * 5 / 4)*radius   0   sin(pi * 5 / 4)*radius
           ];
29     theta_init(:, 7) = [0   0   -radius ];
30     theta_init(:, 8) = [cos(pi * 7 / 4)*radius   0   sin(pi * 7 / 4)*radius
           ];
31     theta_init(:, 9) = [0   radius   0 ];
32     theta_init(:, 10) = [0   cos(pi / 4)*radius   sin(pi / 4)*radius ];
33     theta_init(:, 11) = [0   cos(pi * 3 / 4)*radius   sin(pi * 3 / 4)*radius
           ];
34
35     theta_init(:, 12) = [0   -radius   0 ];
36     theta_init(:, 13) = [0   cos(pi * 5 / 4)*radius   sin(pi * 5 / 4)*radius
            ];
37     theta_init(:, 14) = [0   cos(pi * 7 / 4)*radius   sin(pi * 7 / 4)*radius
           ];
38
39     Acc{i}=Acc{i}';
40
41     %Algoritmo K-means
```

```
42      [~,bel{i},~] = k_means(Acc{i}, theta_init);
43
44  end
45
46  %%Setando o vetor pi de probabilidade inicial de estados
47  for i=1:NumStates
48      if i==1
49          pi_init(i) = 1;
50      else
51          pi_init(i) = 0;
52      end
53  end
54  pi_init = pi_init';
55
56  % Limite de salto de estados
57  jumplimit = 1;
58  % Construindo a matriz inicial de transicao de estados A
59  for i=1:NumStates
60      for j=1:NumStates
61
62          if i==NumStates && j==NumStates
63              A_init(i,j) = 1.0;
64          else if i==NumStates-1 && j==NumStates-1
65                  A_init(i,j) = 0.5;
66              else if i==NumStates-1 && j==NumStates
67                      A_init(i,j) = 0.5;
68                  else if i<=j && i>j-jumplimit-1
69                          A_init(i,j) = 1.0/(jumplimit+1);
70                      else
71                          A_init(i,j) = 0.0;
72                      end
73                  end
74              end
75          end
76
77      end
78  end
79
80
81
82  for i=1:m
83      for j=1:NumStates
84          B_init(i,j) = 1/m;
85      end
86  end
87
88  % Treinando a HMM usando o algoritmo Baum-Welch
```

```
89 [piTrained_1, ATrained_1, BTrained_1, SumRecProbs_1]=...
90     MultSeqTrainDoHMMBWsc(pi_init, A_init, B_init, bel, maxEpoch);
91
92 end
```

Listing A.5 – Rotina de leitura dos dados do acelerômetro

```
1  function [ Acc ] = parse_acc_data( numOfSequencesToCalculate, path_begin )
2  %PARSE_ACC_DATA Ler sequencia de amostras do arquivo
3
4
5  formatSpec = 'x:%f␣y:%f␣z:%f\n';
6  sizeCount = [3 Inf];
7  for i=1:numOfSequencesToCalculate
8      file_path = strcat(path_begin,num2str(i),'.txt');
9      fp = fopen(file_path,'r');
10
11     accData = fscanf(fp,formatSpec,sizeCount);
12     accData = accData';
13
14     Acc{i}(:,1) = accData(:,1);
15     Acc{i}(:,2) = accData(:,2);
16     Acc{i}(:,3) = accData(:,3);
17     fclose(fp);
18
19 end
20
21 end
```

Listing A.6 – Rotina para gerar dados da curva CMC

```
1  function [ rank_t_rate ] = calcRankT( data_matrix, genuine_scores, length )
2
3  % Ordenando linhas da matriz
4  sorted_matrix = sort(data_matrix,2,'descend');
5
6  %Encontrando a posicao (rank) do melhor match para cada linha
7  rank_pos = zeros(1,length);
8  for i=1:length
9      rank_pos(i) = find(sorted_matrix(i,:) == genuine_scores(i), 1);
10 end
11
12 %Calcula a taxa rank-t em percentagem
13 rank_t_rate = zeros(1,length);
14 for i=1:length
15     rank_t_rate(i) = size(find(rank_pos <= i),2)*100/length;
16 end
17
18
```

```
19 end
```

Listing A.7 – Rotina para calcular falsos positivos e falsos negativos

```
1  function [ far,frr,thr_range, eer ] = calcFarFrr( genuine_scores,
       impostor_scores,min_thr,max_thr, ROC_points )
2
3  %Setando o range do threshold
4  step = (max_thr - min_thr)/(ROC_points-1);
5  thr_range = min_thr:step:max_thr;
6
7  %Encontrando FAR and FRR
8  size_impostor =size(impostor_scores);
9  size_genuine = size(genuine_scores);
10 i = 1;
11 for thr=thr_range
12     far(i) = size(find(impostor_scores >= thr))/size_impostor;
13     frr(i) = size(find(genuine_scores <thr))/size_genuine;
14     i = i+1;
15 end
16 %Em porcentagem
17 far = far*100;
18 frr = frr*100;
19
20 %Achando a Equal Error Rate
21 difference = abs(far -frr);
22 [~, ind] = min(difference);
23 eer.thr = thr_range(ind);
24 eer.far = far(ind);
25 eer.frr = frr(ind);
26
27 end
```

Listing A.8 – Rotina para calcular dados das distribuições genuína e impostora

```
1  function [dist] = calcDists( genuine_scores, impostor_scores, Nbins )
2
3  %Calculando a frequencia em cada bin - impostor
4  [imp_count,imp_centers]=hist(impostor_scores,Nbins);
5
6  % Normalizando bins
7  area_curve = trapz(imp_centers,imp_count);
8  imp_count_norm = imp_count/area_curve;
9
10 %Calculando a frequencia em cada bin - genuine
11 [gen_count,gen_centers]=hist(genuine_scores,Nbins);
12
13 % Normalizando bins
14 area_curve = trapz(gen_centers,gen_count);
```

```
15 gen_count_norm = gen_count/area_curve;
16
17
18 %Saida
19 dist.imp_centers = imp_centers;
20 dist.imp_count_norm = imp_count_norm;
21
22 dist.gen_centers = gen_centers;
23 dist.gen_count_norm = gen_count_norm;
24
25 end
```