Avaliação para Trajetórias de Incisões Cirúrgicas com SVM

Assessment for Surgical Incision Trajectories with SVM

Ives Fernando M. S. de Moura [1]
Ronei Marcos Moraes [2]
Liliane dos Santos Machado [3]

Data de submissão: 26/04/2016, *Data de aceite: 11/05/2016*

Resumo: Simuladores de procedimentos cirúrgicos com Realidade Virtual vêm se tornando cada vez mais populares. Um componente importante deste tipo de *software* para a área médica é a capacidade de simular incisões. No contexto computacional, as incisões envolvem três etapas básicas: definição da geometria e topologia, detecção de colisão e deformação. Além disso, em simuladores voltados ao treinamento de estudantes, a capacidade de avaliar o desempenho do usuário é fundamental. Neste trabalho, dois experimentos são feitos, um com dados gerados aleatoriamente e outro com dados gerados a partir de interação do usuário com uma aplicação via mouse, para avaliar a qualidade do corte no que diz respeito à sua trajetória. Ambos os experimentos foram realizados utilizando o método *Support Vector Machine* (SVM). Resultados satisfatórios foram obtidos com a SVM e cinco *kernels*, com taxas de acerto superiores a 70%. Tem-se como principal conclusão, então, que a SVM é um método capaz de avaliar problemas relacionados à trajetória de incisão. Como contribuições do trabalho, têm-se a análise dos componentes da incisão, o levantamento de métodos de avaliação aplicáveis a este problema e o uso da SVM nos dois experimentos conduzidos, que mostrou a aplicabilidade deste método neste contexto.

***P*alavras-chave:** incisão, corte, cirurgia, avaliação, SVM

[1]Laboratório de Tecnologias para o Ensino Virtual e Estatística (LabTEVE), Universidade Federal da Paraíba (UFPB) - João Pessoa, Paraíba, Brasil. {ivesfmoura@gmail.com}
[2]Laboratório de Tecnologias para o Ensino Virtual e Estatística (LabTEVE), Universidade Federal da Paraíba (UFPB) - João Pessoa, Paraíba, Brasil. {ronei@de.ufpb.br}
[3]Laboratório de Tecnologias para o Ensino Virtual e Estatística (LabTEVE), Universidade Federal da Paraíba (UFPB) - João Pessoa, Paraíba, Brasil. {liliane@di.ufpb.br}

Abstract: Virtual Reality simulators for surgical procedures have been gaining popularity. An important component of this type of software for the area of health is the ability to simulate incisions. In the computational context, incisions involve three basic steps: definition of geometry and topology, collision detection and deformation. Furthermore, in simulators aimed for the training of students, the ability to assess user performance is fundamental. In this work, two experiments are conducted, one with randomly generated data e other with data generated through user interaction with an application using a mouse, to assess the quality of the incision with respect to its trajectory. Both experiments were conducted utilizing the Support Vector Machine (SVM) method. Satisfactory results were obtained with the SVM and five kernels, with success rates above 70%. The main conclusion is, therefore, that the SVM is a method capable of assessing problems related to incision trajectory. As contributions of the work, there is the analysis of the incision components, a survey of assessment methods applicable to this problem and the use of the SVM in both experiments, which has shown the applicability of this method in this context.

***K*eywords:** incision, cutting, surgery, assessment, SVM

1 Introdução

A realização de tarefas na área da saúde está condicionada, muitas vezes, ao uso de materiais que são caros ou de difícil acesso, podendo trazer, ainda, riscos à saúde de quem as estiver realizando. Estes fatores servem, então, como motivação para o uso de softwares capazes de realizar ou simular estas tarefas [1]. Programas de computador são empregados nesta área, por exemplo, no treinamento de atuais ou futuros profissionais que desejam adquirir conhecimento sobre procedimentos que podem ser cirúrgicos ou não [2].

No contexto de simuladores para procedimentos cirúrgicos, a possibilidade de realizar incisões (ou cortes) tem papel fundamental, já que é uma etapa presente em grande parte das atividades dessa natureza. A realização destes cortes nos simuladores traz, no entanto, uma série de desafios para o desenvolvimento dessas ferramentas, uma vez que para que se tenha um resultado fisicamente realista é necessário fazer uso de diversas ferramentas matemáticas, físicas e computacionais [3].

Mesmo com esta complexidade associada, os simuladores com Realidade Virtual (RV) vêm se tornando cada vez mais populares no treinamento de estudantes, já que além das vantagens citadas, eles promovem ainda a possibilidade de realizar o mesmo procedimento repetidas vezes, sem a necessidade de subtituir ou repor os materiais utilizados após cada uso. Além disso, métodos para acompanhamento e para a avaliação dos estudantes podem ser integrados a este *software*, podendo fornecer, então, *feedback* aos usuários, para que possam

observar seus erros e ter conhecimento de seu desempenho na realização do procedimento em questão [4].

No contexto de simuladores para procedimentos cirúrgicos, a possibilidade de realizar incisões (ou cortes) tem papel fundamental, já que esta é uma etapa presente em grande parte das atividades dessa natureza. O treinamento para realização de incisões se mostra importante, pois se trata de uma habilidade psicomotora com a qual os alunos têm contato apenas quando já o estão vivenciando em situação prática, o que pode vir a ocasionar insegurança e ansiedade aos alunos, bem como oferecer riscos às partes envolvidas [2]. Com um simulador é possível transmitir para os estudantes a noção do que acontece de fato em uma sala de cirurgia, preparando-os para a prática concreta, uma vez que eles já haverão experimentado situações similares.

Simuladores comerciais, como os listados em [5], que têm suporte a procedimentos com incisão e também avaliação de usuário já são utilizados em contextos hospitalares e universitários atualmente, mas eles possuem a desvantagem de terem alto custo associado. Simuladores com código aberto e sem fins comerciais também existem, mas este tipo de *software* não é de desenvolvimento simples, requerendo extensa pesquisa e equipamentos específicos, o que dificulta a disponibilização dos simuladores para uso livre e com baixo custo.

Será feita, neste trabalho, uma discussão sobre técnicas de incisão utilizadas em simuladores de procedimentos cirúrgicos com RV, mostrando seus requisitos básicos e seu funcionamento. São mostrados também exemplos de softwares deste gênero que já foram desenvolvidos ou continuam em desenvolvimento. Serão expostos, ainda, modelos de decisão que são empregados na avaliação do desempenho de estudantes ao utilizar simuladores deste tipo e as variáveis envolvidas neste processo, uma vez que este é um componente importante em softwares de treinamento. Isto é notável especialmente em se tratando de uma habilidade psicomotora relacionada à área de saúde, como é o caso das incisões, já que, geralmente, os alunos têm contato com este tipo de procedimento apenas quando já estão o vivenciando em situação prática, o que pode gerar ansiedade e insegurança, ocasionando erros e riscos [2]. Neste contexto de avaliação são feitos, por fim, dois experimentos utilizando o modelo *Support Vector Machine* (SVM) para avaliar dados sobre trajetórias de corte. A análise dos componentes da incisão, o levantamento de métodos de avaliação aplicáveis a este problema e o uso da SVM nos dois experimentos conduzidos se destacam como principais contribuições deste trabalho.

2 Corte virtual

Patnaik, Singla e Bansal [6] e outros autores destacam três características essenciais para as incisões: a acessibilidade, a extensibilidade e a segurança. Acessibilidade é relaci-

onada com a capacidade de se alcançar a estrutura anatômica que será explorada e realizar o procedimento necessário. A extensibilidade tem relação com a habilidade de aumentar a incisão em uma direção que facilite a operação, mas não danifique as funções do corpo. Finalmente, segurança neste contexto se relaciona a manutenção das funções corporais do paciente e o cuidado com a vizinhança do local da incisão. É importante, então, que sistemas computacionais que planejam lidar com a simulação de cortes sejam capazes de representar estes aspectos em seu mundo virtual.

A simulação de técnicas de incisão com RV para softwares voltados ao uso na área de saúde deve se preocupar em ser eficiente, robusta e realista, além dos aspectos de saúde já citados. Para isso, três componentes se destacam na implementação do chamado corte virtual: a detecção de colisão entre o intrumento de corte e o objeto a ser cortado, a simulação do corpo como um objeto deformável (com características de elasticidade) e, finalmente, a atualização da representação geométrica e topológica do modelo utilizado para comportar o corte [3].

Na maioria das vezes, os objetos tridimensionais vistos no computador ao utilizar um simulador são representados internamente como um conjunto de polígonos ou poliedros (triângulos ou tetraedros, em geral) chamados de primitivas. A conformação destes objetos geométricos no espaço, isto é, o posicionamento deles e sua vizinhança, forma a topologia do objeto. O corte é, então, uma operação topológica que rearranja as primitivas, subdivindo-as e movimentando-as para gerar uma lacuna no objeto. Os instrumentos de corte (bisturis, lâminas, serras, etc.) também serão representados no simulador como objetos compostos por primitivas. Para saber, então, quando realizar o corte, o sistema deve ser capaz de detectar quando e onde as primitivas do instrumento e do órgão que será cortado estão em contato. Esta tarefa recebe o nome de detecção de colisão [7]. Finalmente, ao tocar com o instrumento de corte no órgão e ao realizar o corte, é esperado que haja contração ou relaxamento dos tecidos envolvidos no processo, já que os tecidos do corpo humano apresentam elasticidade. À simulação destes fenômenos físicos, dá-se o nome de deformação

O procedimento de corte tem início a partir do momento em que há colisão entre o instrumento de corte e o órgão e é finalizado quando esta colisão deixa de ocorrer. À medida que o instrumento é movimentado, a modificação na topologia do órgão é feita e a deformação é aplicada, conforme mostrado no diagrama da Figura 1, elaborado a partir do exposto em [8] e [9]. Outros fenômenos podem ser acrescidos à simulação para aumentar seu realismo, como o uso dos chamados dispositivos hápticos, que fornecem retorno de força ao usuário, para que ele possa sentir de fato o que está tocando no mundo virtual; e a aplicação de técnicas de dinâmica de fluidos, para a simulação de sangramentos.

Do ponto de vista computacional, o corte é uma operação bastante custosa e, por isso, várias abordagens são utilizadas em sua implementação, variando a forma como os três componentes básicos aqui descritos são tratados. Uma discussão mais aprofundada sobre este

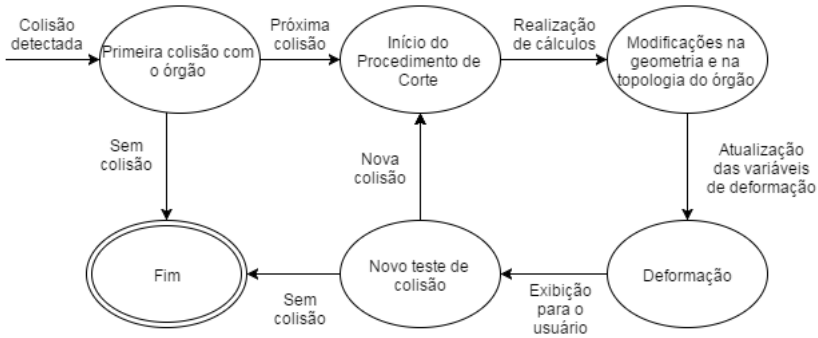

Figura 1. Diagrama do procedimento de corte

assunto pode ser encontrada em [10] e [8].

3 Simuladores de Procedimentos Cirúrgicos com RV

Existe, na literatura, uma série de exemplos de simuladores desenvolvidos para diversos procedimentos cirúrgicos diferentes. Serão citados aqui alguns simuladores cirúgicos com suporte a procedimento de incisão, destacando suas características em relação à forma como realizam o corte.

Com relação à representação geométrica e à topologia dos objetos tridimensionais do simulador, é convencional utilizar objetos compostos por triângulos ou tetraedros. Existem, no entanto, simuladores que fogem dessa convenção, utilizando outras representações para seus objetos. Um exemplo disto pode ser encontrado em [11], que representa seus objetos como nuvens de nós, computando o corte a partir do chamado critério de visibilidade, utilizando o método *level set*.

Já no que diz respeito à detecção de colisão, existe enorme variabilidade dentre os simuladores encontrados. Esta característica é esperada, no entanto, já que a detecção de colisão é completamente dependente do tipo de primitiva utilizado (tanto nos órgãos quanto nos intrumentos, que não necessariamente utilizam o mesmo tipo), existindo ainda diversas estratégias para a otimização desta tarefa, em vista de seu custo computacional. Uma discussão aprofundada sobre detecção de colisão pode ser encontrada em [12]. Como exemplos de diferentes abordagens para detecção de colisão, podemos citar [4], que se utilizou de método generalizado utilizando cilindros; [13], que se utiliza da distância do instrumento ao plano de corte e métodos analíticos; e [10], que fazem uso de uma abordagem baseada na topologia descrita por eles, chamada de Elementos Finitos Contínuos, que se utiliza de primitivas

hexaédricas.

Finalmente, no tocante à deformação, duas técnicas específicas são mais comumente encontradas: o modelo Massa-Mola, exemplificado em [13], e a técnica de Elementos Finitos, que pode ser encontrada em [14]. Há, no entanto, outras técnicas disponíveis, como visto em [15], que utilizam uma técnica de dinâmica baseada em posição; e em [16], que utilizam a chamada *Free-Form Deformation* (FFD).

Um grande número de simuladores com suporte a incisão existe na literatura, empregando diversas técnicas para alcançar seus objetivos. É interessante observar que, apesar de existir uma predominância de simulações para procedimentos laparoscópicos, ainda assim é possível encontrar simuladores que tratam de procedimentos que requerem incisões em maior escala ou que não comportam laparoscopia, como pode ser visto em [17] e [18], por exemplo.

4 Treinamento e Avaliação de Incisões com Simuladores

As possibilidades de reduzir custos e riscos, substituir materiais caros ou de difícil acesso e de repetir a execução de um determinado procedimento várias vezes sem a necessidade de utilizar novos materiais a cada vez ajudam a tornar o uso de simuladores para o treinamento de estudantes atrativo [19]. Outro aspecto interessante neste contexto é a possibilidade de promover a colaboração entre profissionais e estudantes de diferentes áreas durante o treinamento, através dos chamados Ambientes Virtuais Colaborativos [20].

A partir das interações dos usuários com o simulador é possível extrair diversas informações que podem ser utilizadas posteriormente para realizar a avaliação do desempenho do indivíduo [19]. No caso das incisões, o sistema pode coletar informação sobre posição do instrumento de corte, força aplicada, extensão do corte, dentre outras, que podem ser então comparadas com informações presentes na literatura da área ou obtidas através de conhecimento especialista. Essas informações, que neste contexto são chamadas de métricas, constituem uma limitação para os sistemas de avaliação, uma vez que elas dificilmente são bem-definidas e seguidas rigidamente no mundo real [21].

Esta característica de inexistência ou negligência de métricas introduz grande incerteza em relação aos valores medidos a partir da simulação, pois torna-se complexo definir o que é certo ou errado com precisão. Além disso, é possível tratar também de variáveis qualitativas, por exemplo, questões estéticas relacionadas ao corte. Vê-se, então, que além de não existirem métricas universalmente aceitas, os diferentes tipos de variáveis existentes no problema também influenciam na avaliação do usuário. É necessário, portanto, escolher um modelo de decisão que seja capaz de se adequar a essas restrições.

Paiva et al. [22] descrevem quatro modelos de decisão comumente utilizados em simuladores com RV. A Tabela 1 traz estas informações de maneira adaptada, incluindo o

Tabela 1. Diferentes modelos de decisão, suas descrições e tipos de variáveis suportados. Adaptada de [22] (p. 5)

Modelo de Decisão	Descrição
Lógica Clássica	Utilizada quando o problema possui estados com relação de causalidade bem definida
Lógica *Fuzzy*	Utilizada quando se possui dados com imprecisão ou incerteza
Sistemas Especialistas Baseados em Regras	Utilizados para modelar cenários definidos a partir de conhecimento de especialistas humanos
Árvore de Decisão	Utilizada para organizar o conhecimento a partir do ganho de informação associado aos atributos do problema
Support Vector Machine	Utilizada para agrupar dados em conjuntos a partir do uso de estruturas chamadas hiperplanos

modelo *Support Vector Machine*, o qual será discutido posteriormente neste trabalho. A avaliação do corte pode tratar de variáveis ordinais, intervalares ou nominais, dependendo do que for levado em consideração para classificar a incisão como boa ou não. Além disso, como existe imprecisão nas variáveis, elas podem ser modeladas como variáveis *fuzzy*, que tratam exatamente deste tipo de problema [23].

Quatro métricas são definidas em [24] para avaliar performance em treinamento de laparoscopia: o tempo para completar a tarefa, o comprimento do caminho percorrido pelo instrumento de corte, o volume da menor caixa que contém todas as amostras capturadas com o sensor na ponta do instrumento utilizado e o esforço de controle, que tem relação com as forças que o usuário gerou durante a simulação, as quais são calculadas analiticamente a partir dos dados obtidos. A partir destas ideias, é possível delinear métricas similares para a avaliação de tarefas de incisão.

Com a técnica de detecção de colisão utilizada, serão obtidas primitivas, as quais podem ser utilizadas então para construir o caminho do corte, sendo possível extrair, portanto, informações sobre comprimento e trajetória, por exemplo. Além disso, de acordo com o dispositivo que se utilizar para interagir com o sistema, será possível obter também dados de profundidade, força, aceleração, etc. Se a simulação for capaz de exibir e processar não apenas o órgão que será cortado, como também sua vizinhança, é possível avaliar aspectos de segurança no corte, no sentido de verificar se o usuário atingiu ou não regiões que não deveria. Podem-se incluir ainda outros aspectos, como os estéticos ou os de biossegurança, mas é importante perceber que a cada variável incluída na avaliação, o sistema se torna mais complexo.

Além da complexidade, tem-se o problema de definição de métricas, como já citado. Por mais que haja definições ou guias para a realização de incisões específicas, como pode ser visto em [6], os valores contidos na literatura da área não são necessariamente aceitos pela comunidade, podendo haver divergências entre as noções de corretude entre diferentes especialistas na área.

A SVM, utilizada em [24] como forma de avaliar a performance do usuário em procedimento laparoscópico com base nas quatro métricas definidas apresentou bons resultados. Devido à similaridade entre as métricas deste trabalho e as envolvidas na avaliação de incisões, a SVM se mostra uma candidata interessante para esta tarefa.

A SVM é um método proposto originalmente na década de 60, mas que evoluiu ao longo das décadas, sendo sua versão atualmente mais conhecida, referida como *soft margin*, descrita em [25].A ideia inicial da SVM é que dadas duas classes de dados, é possível construir um hiperplano que as separe, de forma que novos dados possam ser classificados de acordo com sua posição relativa ao hiperplano. Esta ideia pode ser então estendida para situações em que há mais de duas classes (conjuntos de hiperplanos) e, ainda, para casos em que os dados não são linearmente separáveis, através de um artifício chamado de *kernel*.

A versão linear da SVM procura escolher o melhor hiperplano de separação a partir do conceito de margens. As margens do hiperplano são as distâncias do plano aos dados mais próximos dele em cada classe. A SVM busca, então, maximizar essa distância mínima, fazendo o problema se tornar um clássico problema de otimização [26], representado pela Equação 1, onde m é o número de elementos do conjunto de treino, $cost_0$ e $cost_1$ são as funções de custo associadas à pertinência de uma amostra à classe ou não, $y^{(i)}$ é a variável de saída para a amostra i, $x^{(i)}$ é o vetor de variáveis de entrada para a amostra i, θ^T é o vetor de parâmetros transposto e o termo final é o chamado termo de regularização, que é utilizado para lidar com o problema de *overfitting* [27].

$$min_\theta C = \sum_{i=1}^{m}[y^{(i)}cost_1(\theta^T x^{(i)}) + (1 - y^{(i)})cost_0(\theta^T x^{(i)})] + \frac{1}{2}\sum_{i=1}^{n}\theta_j^2 \qquad (1)$$

Para tratar, por fim, de casos que não são linearmente separáveis, utilizam-se os *kernels*, que podem ser entendidos como métodos para alterar a dimensionalidade dos dados, mapeando-os para outro espaço vetorial, onde seja possível separá-los linearmente [28]. Concretamente, a hipótese da SVM (que computa a Equação 1 para decidir a que classe uma amostra pertence) pode ser reescrita como a soma do vetor de parâmetros θ multiplicado por um novo vetor de características f, como mostrado na Equação 2. Cada elemento do vetor de características pode ser entendido, então como uma função de similaridade (f_i) entre os valores de entrada (x) e as chamadas *landmarks* (l_i), que são pontos definidos no espaço do problema como representativos das classes, como mostrado na Equação 3. Estas funções de

similaridade são chamadas, finalmente, de *kernels* [29].

$$h_\theta(x) = \theta_0 + \theta_1 f_1 + \theta_2 f_2 + ... + \theta_n f_n \tag{2}$$

$$f_i = similaridade(x, l_i) \tag{3}$$

Uma série de conceitos matemáticos e estatísticos estão envolvidos na concepção das SVMs, trazendo várias vantagens consigo, como o suporte a dados não-linearmente separáveis e a alta dimensionalidade de dados. Este modelo é explorado de forma mais detalhada em [25].

5 Avaliação de trajetórias de corte utilizando SVMs

A partir da técnica de detecção de colisão utilizada, em conjunto com um dispositivo de interação, é possível obter uma lista com as primitivas intersectadas ao longo do movimento realizado. Estas primitivas, que podem ser pontos, triângulos, etc., podem ser então utilizadas para discretizar o caminho de corte. Define-se, então, a trajetória do corte como um conjunto de primitivas, no qual se destacam as primitivas inicial e final do corte, e todos os outros elementos do conjunto, quando conectados, formam um caminho coeso entre as duas extremidades. Coeso, neste contexto, significa que o caminho tem formato similar ao seu equivalente contínuo e ao que se esperaria no mundo real, não havendo desvios significativos.

Se for considerado um procedimento qualquer que exija uma incisão retilínea, o caminho ideal seria, claramente, uma reta que ligasse estes dois pontos, como visto na Figura 2. Em uma situação real, no entanto, dificilmente este caminho de corte será obtido, pois podem existir variações no posicionamento dos pontos inicial e final por parte de quem estiver realizando a incisão e, também, raramente será possível produzir uma incisão perfeitamente retilínea a mão livre. É preciso, então, especificar o que é considerado um bom ponto inicial (ou ponto final) ou não. Uma abordagem possível para isso é definir regiões em torno do ponto, conforme pode ser visto na Figura 2. A região mais interna conteria os pontos considerados bons, a do meio conteria os pontos considerados aceitáveis e a mais externa, os pontos considerados ruins. Pontos fora destas regiões seriam desconsiderados, pois não refletiriam um comportamento real, já que espera-se que um usuário, ao utilizar o simulador, esteja fazendo esforço para permanecer no caminho correto. Com esta primeira ideia, cortes bons seriam aqueles que ligam pontos em regiões boas, cortes aceitáveis são aqueles em que pelo menos um dos pontos está na região aceitável e cortes ruins seriam aqueles em que pelo menos um dos pontos está na região ruim.

O problema com esta abordagem é que ela não considera casos de fronteira, isto é, casos em que poucos pontos se localizam fora de determinada região, mas a maioria restante

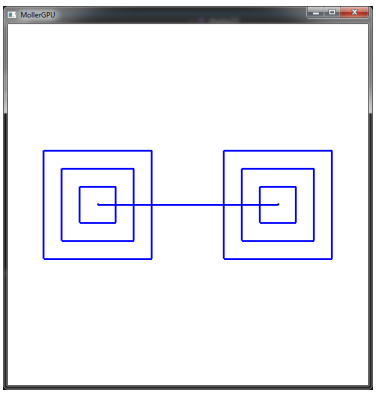

Figura 2. Exemplo de corte retilíneo ideal e regiões de classificação para os pontos inicial e final

está dentro dela acabam recebendo classificação indevida por causa da minoria que ficou de fora. Além disso, a noção de trajetória também acaba perdida, já que coloca-se ênfase apenas nos pontos incial e final, tornando os intermediários irrelevantes. Para corrigir este problema, deve-se estender as regiões para englobar não apenas os pontos extremos, mas todo o corte, criando-se então regiões de cortes bons, aceitáveis ou ruins. Além disso, deve-se considerar a que regiões os pontos coletados pertencem, atribuindo a categoria da maioria dos pontos ao corte. É importante observar, ainda, que esta ideia pode ser aplicada também a trajetórias com outros formatos, desde que se delimite corretamente as regiões boa, ruim e mediana e que se amostre uma quantidade suficiente de pontos.

De posse dessas duas ideias, pode-se construir dois experimentos, com o objetivo de avaliar se há a possibilidade de classificar cortes nas categorias definidas (Bom, Aceitável ou Médio e Ruim). As próximas seções deste artigo descrevem a metodologia utilizada e os resultados obtidos.

6 Metodologia

Com base nas duas ideias desenvolvidas anteriormente, dois experimentos foram realizados, com o objetivo de avaliar a viabilidade de utilizar as representações explicadas para a avaliação de cortes, no que diz respeito à trajetória deles.

Para o primeiro experimento, definiu-se que o corte seria retilíneo, partindo do ponto (10,10) para o ponto (20,10). Definiu-se, ainda, que a região considerada boa teria raio 1,0 ao redor dos pontos, a região aceitável teria raio 2,0 e a região ruim, 3,0. Foram gerados, então,

50 pontos aleatórios dentro de cada uma das regiões, tanto para o ponto inicial, quanto para o final. Após isso, 60 pares de pontos foram gerados para cada categoria e classificados de acordo com as regiões dos pontos, formando então o primeiro conjunto de dados, com 180 amostras. Para integrar o banco de dados, foi calculada a equação da reta para cada par de pontos, e pontos intermediários foram obtidos, gerando, finalmente, múltiplas retas. O banco de dados consiste então de 60 amostras para cada classe, e cada amostra representa um par de pontos aleatoriamente gerado dentro das regiões especificadas e 9 pontos intermediários entre eles.

No segundo experimento, o corte retilíneo foi mantido, mas a forma como os dados foram gerados foi diferente. Os dados agora passaram a ser capturados a partir da interação do usuário com a aplicação, via mouse, o que representaria a segunda ideia explorada na seção anterior, capturando os pontos intermediários diretamente a partir da interação do usuário. A aplicação capturava a posição do mouse no espaço da imagem (posição dos pixels) a partir do momento em que o usuário clicasse em algum ponto (sendo este seu ponto inicial) até que ele parasse de pressionar o botão (momento este em que seria registrado o ponto final). Como o volume de dados gerado dessa maneira é grande, foi feita posteriormente uma amostragem de 11 pontos por corte realizado, formando finalmente o segundo conjunto de dados. Este conjunto, assim como o primeiro, também contém 60 amostras de cada classe, tendo então um total de 180 amostras, mas agora os dados são classificados de acordo com a região a que a maioria dos pontos registrados durante a interação pertence, isto é, se todos os pontos estiverem na região considerada boa, a trajetória será considerada boa; se houver pontos apenas da região mediana ou tanto da mediana quanto da boa, o corte é considerado mediano e, finalmente, aquelas trajetórias que contiverem pontos da região ruim, sejam elem exclusivos ou em conjunto com pontos das outras regiões, são consideradas ruins. Todo o conjunto de dados desse experimento foi gerado a partir da interação do desenvolvedor do programa com a aplicação, estando este ciente dos parâmetros utilizados e da forma como deveria realizar o procedimento, evitando então a inclusão de casos que não refletem a realidade, como trajetórias em zigue-zague, por exemplo.

Para a criação de ambos os conjuntos de dados, foi feito um programa, utilizando a linguagem de programação C++. Na geração do primeiro conjunto, destaca-se o uso da biblioteca *random* da linguagem C++, em especial os componentes *random_device*, que busca gerar números aleatórios de forma não-determinística através de processos estocásticos, e a classe *uniform_real_distribution*, que utiliza o dispositivo para gerar números aleatórios com distribuição uniforme real. Já para gerar o segundo conjunto de dados, foi utilizada a biblioteca gráfica OpenGL e a biblioteca de utilitários GLUT, para que fosse possível fornecer a visualização dos pontos e das regiões de corte e gerenciar a interação do usuário com o ambiente, salvando os pontos obtidos com o movimento do mouse.

Construídos os conjuntos de dados, partiu-se para a avaliação deles com o uso de

um modelo de decisão específico, neste caso, as *Support Vector Machines* (SVMs). Para ter acesso a esse modelo, utilizou-se o software Weka (versão 3.6.12), que conta com duas implementações deste método, a libSVM e a SMO, cujas implementações das SVMs diferem pelos métodos numéricos que utilizam internamente. Os algoritmos implementados por ambos os métodos são explicados em [26] e [30], respectivamente.

Para avaliar os resultados obtidos ao fim de cada execução da SVM, utilizou-se o coeficiente Kappa, que é fornecido pelo Weka no relatório de cada execução. O coeficiente Kappa é uma medida de concordância, bastante utilizada na literatura de avaliação como forma de medir o desempenho em relação à quantidade de casos corretamente avaliados, tendo sido proposto por Cohen (1960). Este coeficiente assume valores entre 0 e 1 e, quanto mais próximo de um, melhor o modelo é capaz de descrever os dados, gerando avaliações mais precisas. Com o objetivo de encontrar que versão da SVM melhor se adaptaria a estes dados, 7 *kernels* diferentes foram utilizados com cada conjunto de dados, 3 com o método libSVM (linear, polinomial e RBF) e 4 com o SMO (polinomial, polinomial normalizado, Puk e RBF), sendo registrados os coeficientes Kappa e as taxas de acerto para cada execução, como mostrado nos resultados.

7 Resultados e Discussão

O modelo SVM foi empregado nos dois experimentos descritos anteriormente. Para cada experimento 7 *kernels* diferentes foram utilizados por meio do *software* Weka e suas implementações de SVM (libSVM ou SMO). Em todos os testes foi utilizada *cross-validation* com 10 *folds* a partir dos bancos de dados descritos na Metodologia.

O primeiro algoritmo executado foi a libSVM. Todos os testes realizados com este algoritmo utilizaram o tipo C-SVC de SVM, que é utilizado como classificador com custo $C = 1$. O primeiro *kernel* utilizado foi o chamado *Radial Basis Function* (RBF), que funciona como uma medida de similaridade entre os dados, sendo baseado na distância Euclidiana [31] e cuja formulação pode ser vista na Equação 4. Cinco valores foram utilizados para o parâmetro γ, a fim de obter melhor adequação: 0 (interpretado pelo Weka como $\frac{1}{maxIndex}$), 1, 0,1, 0,01, 0,5. Para o primeiro experimento, este método obteve coeficiente Kappa 0,8417, com 89,4% de taxa de acerto com $\gamma = 0$, 0,8083 e 87,22% com $\gamma = 1$, 0,8833 e 92.22% com $\gamma = 0,1$, 0,7750 e 85% com $\gamma = 0,01$ e 0,8750 e 91,66% com $\gamma = 0,5$. Já para o segundo experimento, foi obtido um Kappa de 0,03 com 35,6% de taxa de acerto para ($\gamma = 0$), 0 e 33,33% para ($\gamma = 1$), 0,07 e 37,78% para ($\gamma = 0,1$), 0,08 e 38,89% para ($\gamma = 0,01$) e 0,1 e 40% para ($\gamma = 0,5$).

$$f_i = \exp^{-\gamma|u-v|^2} \tag{4}$$

Continuando com a libSVM, mas agora com o *kernel* linear (que significa, concre-

tamente, que não se está utilizando um *kernel*, mas resolvendo o problema de otimização original do método, mostrado na Equação 1), foram obtidos Kappas 0,3917 e 0,2, com taxas de acerto 59,4% e 46,7% para os experimentos 1 e 2, respectivamente.

O último *kernel* executado junto do método libSVM foi o polinomial, que assim como o RBF também é utilizado como medida de similaridade, sendo a Equação 5 sua representação na libSVM, com c_0 representando um coeficiente constante e d o grau do polinômio. O valor utilizado para c_0 foi 1, para d foi 3 e os valores de γ permanecem os mesmos descritos para o RBF. As taxas de acerto passaram a ser 81.1% ($\gamma = 0$), 60% ($\gamma = 1$), 70,55% ($\gamma = 0,1$), 86,66% ($\gamma = 0,01$) e 63,33% ($\gamma = 0,5$) para o primeiro experimento e 73,89%, 73,33%, 72,22%, 73,33% e 73,33, respectivamente, para o segundo, com coeficientes Kappa de 0,7167, 0,4, 0,5583, 0,8 e 0,45 para os testes com o primeiro experimento e 0,6083, 0,60, 0,5833, 0,60 e 0,60 respectivamente, para o experimento 2.

$$f_i = (\gamma * u^T * v + c_0)^d \tag{5}$$

Passando-se, então, para o método SMO, temos a execução de quatro outros *kernels* com a constante de custo sendo mantida em 1, assim como foi feito com a libSVM. O primeiro *kernel* executado foi o polinomial fornecido pelo Weka que, diferente do anterior, não permite variação de γ, apenas de grau, tendo esta opção recebido valor 3. O primeiro experimento gerou os valores 0,7083 e 80,55% como Kappa e taxa de acerto, respectivamente, e o segundo experimento gerou os valores 0,6333 e 75,56%.

O próximo *kernel* executado em conjunto com o SMO foi o *kernel* polinomial normalizado, que tem as mesmas características do *kernel* polinomial anterior, mas que normaliza os vetores de dados para a execução dos cálculos. Para o primeiro experimento obteve-se Kappa de 0,5417 e a taxa de acerto 69,44%; e para o segundo experimento forneceu coeficiente 0,5083 e taxa 67,22%.

Em seguida, foi utilizado o *kernel* Puk junto do método SMO. Este *kernel* utiliza a função Pearson VII, que tem formato similar à função normal, sendo representada pela Equação 6 [32]. Dois parâmetros podem ser variados neste kernel, σ e ω. Foram feitos os seguintes pares para teste: (1,1), (0,1, 1), (1, 0,1) e (0,1, 0,1). Ao executar o experimento 1, o Weka retornou em seu relatório o valor de coeficiente Kappa 0,8833 e taxa de acerto de 92,2% para o primeiro par de parâmetros, 0,8667 e 91,11% para o segundo, 0,8917 e 92,78% para o terceiro e 0,85 e 90% para o último par. Já ao executar o segundo experimento, o coeficiente Kappa obtido foi 0,8917, com taxa de acerto de 92,8% para o primeiro par, 0,8250 e 88,33% para o segundo, 0,7917 e 86,11% para o terceiro e 0,8333 e 88,89% para o quarto par.

$$f_i = \frac{1}{[1 + (2\sqrt{\|u - v\|^2}\frac{\sqrt{2^{(1/\omega)}-1}}{\sigma})^2]^\omega} \tag{6}$$

Finalmente, o último *kernel* utilizado foi o RBF, mas assim como o polinomial, com o SMO utilizou-se a versão fornecida pelo Weka, que também permite alterar os valores para γ e, por isso, foram utilizados os mesmos valores estabelecidos anteriormente para estes novos testes, com exceção do 0, já que este *kernel* não faz a mesma interpretação que o seu equivalente na libSVM faz. Foram observados coefiecientes Kappa 0,8417 ($\gamma = 1,0$), 0,60 ($\gamma = 0,1$), 0,175 ($\gamma = 0,1$) e 0,8250 ($\gamma = 0,5$) no primeiro experimento e 0,8750, 0,6167, 0,15 e 0,85, respectivamente, no segundo experimento. As taxas de acerto observadas no primeiro experimento foram 89,44%, 73,33%, 45% e 88,33% respectivamente. Já no segundo experimento, foram observadas, respectivamente, as seguintes taxas de acerto: 91,67%, 74,44%, 43,3% e 90%. Os melhores resultados obtidos com cada *kernel* para cada experimento são sintetizados na Tabela 2.

A Tabela 2 inclui ainda os resultados obtidos com os algoritmos J48 e JRip disponíveis também no *software* Weka, que representam os modelos árvore de decisão e lógica clássica, respectivamente. Com o algoritmo J48 foi obtida taxa de acerto de 96,11% e Kappa 0,9417 para o experimento 1 e taxa 87,78% e Kappa 0,8167 para o experimento 2. Já o algoritmo JRip apresentou taxas de acerto 86,11% e 88,33% e coeficientes Kappa 0,7917 e 0,8250 para os experimentos 1 e 2, respectivamente.

Dentre todos os diferentes *kernels* utilizados, o Puk foi o que retornou melhor coeficiente Kappa para ambos os experimentos. É interessante observar também o mau desempenho do *kernel* RBF com a libSVM no segundo experimento, o que indica que a implementação do *kernel* RBF utilizada pelo SMO é mais estável, em relação a estes conjuntos de dados. Nota-se, no entanto, que este *kernel* parece ser adequado para o experimento 1, principalmente no resultado obtido com a libSVM. Isto é esperado, uma vez que é possível separar os pontos iniciais e finais em regiões, e as distâncias entre tais regiões são bem-definidas. Outro *kernel* que obteve bons resultados em ambos os experimentos foi o polinomial, mas ele não se adequa a sistemas de avaliação em tempo real, pois ele demorou vários minutos para terminar seus cálculos. O *kernel* linear não obteve resultados satisfatórios em nenhum dos dois experimentos, o que era esperado, uma vez que os dados utilizados, especialmente no experimento 2, são não-lineares. É notável também que quando comparada a outros métodos, especialmente no experimento 2, a SVM com *kernel* Puk ainda apresenta melhores resultados, confirmando a ideia de que a SVM é adequada para a tarefa de avaliação de corte, em especial de trajetória.

8 Considerações Finais

A partir do exposto neste artigo, é possível perceber que tanto o processo de corte virtual quanto o de avaliação de usuários de simuladores são complexos, sendo a integração de ambos em um único sistema uma tarefa árdua. É necessário, então, fazer o planejamento e

Tabela 2. Coeficiente Kappa e taxa de acerto obtidos com os métodos libSVM e SMO e 7 diferentes kernels

	Coeficiente Kappa	Taxa de Acerto
libSVM com kernel RBF		
Experimento 1	0,8833	92,22
Experimento 2	0,1000	40,00
libSVM com kernel linear		
Experimento 1	0,3917	59,40
Experimento 2	0,2000	46,70
libSVM com kernel polinomial		
Experimento 1	0,8000	86,66
Experimento 2	0,6083	73,89
SMO com kernel polinomial		
Experimento 1	0,7083	80,55
Experimento 2	0,6333	75,56
SMO com kernel polinomial normalizado		
Experimento 1	0,5417	69,44
Experimento 2	0,5083	67,22
SMO com kernel Puk		
Experimento 1	0,8917	92,78
Experimento 2	0,8917	92,80
SMO com kernel RBF		
Experimento 1	0,8417	89,44
Experimento 2	0,8750	91,67
J48		
Experimento 1	0,9417	96,11
Experimento 2	0,8167	87,78
JRip		
Experimento 1	0,7917	86,11
Experimento 2	0,8250	88,33

o levantamento de requisitos de ambos com atenção, para que o processo de implementação possa ser conduzido de maneira coordenada.

A escolha da técnica de corte que será utilizada (incluindo a escolha da representação geométrica, da técnica de detecção de colisão e da técnica de deformação) depende bastante de procedimento cirúrgico que será simulado e das características desejadas no sistema, como realismo físico, alto desempenho, alta qualidade gráfica, etc. Já a escolha do modelo de decisão que será utilizado na avaliação dos usuários deve ser feita com base nos parâmetros que serão utilizados para avaliar. No caso específico das incisões, os sistemas baseados em lógica *fuzzy* podem se mostrar adequados para esta tarefa, uma vez que eles são capazes de descrever os tipos de variáveis presentes no problema e de acomodar as imprecisões. Apesar disso, outros modelos que não utilizam lógica *fuzzy* também podem ser capazes de se adequar ao problema da avaliação do corte, como é o caso das SVMs analisadas aqui. Apesar de dois *kernels* terem retornado resultados insatisfatórios (com coeficientes Kappa baixos), o método libSVM com *kernel* polinomial e com o *kernel* RBF (para o experimento 1) e o método SMO com os *kernels* Puk, polinomial e RBF se mostram bons candidatos para uso em tarefas de classificação relacionadas ao corte, ao menos no que diz respeito à trajetória do corte, sendo esta análise a principal contribuição deste trabalho.

É importante notar que a maioria dos simuladores apresentados neste artigo não apresentam suporte à avaliação de usuário, sendo muitos nem mesmo validados por especialistas. Estes são aspectos importantes nesse contexto, especialmente pela intenção de se utilizar os simuladores para treinar futuros profissionais. É compreensível, no entanto, que haja essa lacuna nos trabalhos da área, uma vez que a inexistência de métricas universalmente aceitas pela comunidade científica representa um grande empecilho para a realização destas tarefas, o que representa, inclusive, uma limitação para este trabalho.

Finalmente, destacam-se como trabalhos futuros a oportunidade de testar o poder de avaliação das SVMs com outras variáveis importantes do corte, como a profundidade, a força aplicada, etc., a comparação das classificações obtidas com a SVM com outras obtidas utilizando outros modelos, o uso de dados obtidos a partir de outros dispositivos de interação para geração dos conjuntos de teste e a validação destes dados por especialistas, que poderiam transformar os dados em algo que se assemelha mais à realidade.

Contribuição dos autores:

- Ives Fernando Martins Santos de Moura: Responsável pela execução dos experimentos descritos no artigo. Participou ainda da escrita de todas as seções do texto.

- Ronei Marcos de Moraes: As seções de avaliação, do método SVM e elaboração dos experimentos foram construídas com contribuições deste autor. Também foi realizada a

revisão final do texto.

- Liliane dos Santos Machado: A introdução do artigo e as seções associadas aos conceitos de corte virtual e simuladores com Realidade Virtual foram elaboradas com contribuições desta autora.

Referências

[1] Moraes RM, Machado LS. Assessment Systems for Training Based on Virtual Reality: A Comparison Study. SBC Journal on 3D Interactive Systems. 2012;3(1):9–16.

[2] Moraes RM, Rocha AV, Machado LS. Intelligent assessment based on Beta Regression for realistic training on medical simulators. Knowledge-Based Systems. 2012;32:3–8.

[3] Wu J, Westermann R, Dick C. Physicallybased simulation of cuts in deformable bodies: A survey. Eurograph State-of-the-Art Report. 2014;2.

[4] Pan JJ, Chang J, Yang X, Liang H, Zhang JJ, Qureshi T, et al. Virtual reality training and assessment in laparoscopic rectum surgery. The International Journal of Medical Robotics and Computer Assisted Surgery. 2014;.

[5] SenseGraphics. References; 2016. http://sensegraphics.com/references/.

[6] Patnaik V, Singla RK, Bansal V. Surgical incisions - their anatomical basis Part IV - abdomen. J Anat Soc India. 2001;50(2):170–8.

[7] Tang M, Manocha D, Lin J, Tong R. Collision-streams: fast gpu-based collision detection for deformable models. In: Symposium on interactive 3D graphics and games. ACM; 2011. p. 63–70.

[8] Bruyns CD, Senger S, Menon A, Montgomery K, Wildermuth S, Boyle R. A survey of interactive mesh-cutting techniques and a new method for implementing generalized interactive mesh cutting using virtual tools. The journal of visualization and computer animation. 2002;13(1):21–42.

[9] Choi KS. Interactive cutting of deformable objects using force propagation approach and digital design analogy. Computers & Graphics. 2006;30(2):233–243.

[10] Wu J, Westermann R, Dick C. Real-time haptic cutting of high resolution soft tissues. Studies Health Technol Inform(Proc Medicine Meets Virtual Reality 2014). 2014;196:469–475.

[11] Jin X, Joldes GR, Miller K, Yang KH, Wittek A. Meshless algorithm for soft tissue cutting in surgical simulation. Computer methods in biomechanics and biomedical engineering. 2014;17(7):800–811.

[12] Ericson C. Real-time collision detection. CRC Press; 2004.

[13] Zhang Y, Yuan Z, Ding Y, Zhao J, Duan Z, Sun M. Real Time Simulation of Tissue Cutting Based on GPU and CUDA for Surgical Training. In: Biomedical Engineering and Computer Science (ICBECS), 2010 International Conference on. IEEE; 2010. p. 1–4.

[14] Lu Z, Arikatla VS, Han Z, Allen BF, De S. A physics-based algorithm for real-time simulation of electrosurgery procedures in minimally invasive surgery. The International Journal of Medical Robotics and Computer Assisted Surgery. 2014;10(4):495–504.

[15] Pan J, Bai J, Zhao X, Hao A, Qin H. Real-time haptic manipulation and cutting of hybrid soft tissue models by extended position-based dynamics. Computer Animation and Virtual Worlds. 2015;26(3-4):321–335.

[16] Sela G, Subag J, Lindblad A, Albocher D, Schein S, Elber G. Real-time haptic incision simulation using FEM-based discontinuous free-form deformation. Computer-Aided Design. 2007;39(8):685–693.

[17] Ho AK, Alsaffar H, Doyle PC, Ladak HM, Agrawal SK. Virtual reality myringotomy simulation with real-time deformation: Development and validity testing. The Laryngoscope. 2012;122(8):1844–1851.

[18] Xia P, Lopes AM, Restivo MT. Virtual reality and haptics for dental surgery: a personal review. The Visual Computer. 2013;29(5):433–447.

[19] Moraes RM, Machado LS. Psychomotor skills assessment in medical training based on virtual reality using a Weighted Possibilistic approach. Knowledge-Based Systems. 2014;70:97–102.

[20] Paiva PVF, Machado LS, Gondim Valença MM. A Virtual Environment for Training and Assessment of Surgical Teams. In: XV Symposium on Virtual and Augmented Reality (SVR). IEEE; 2013. p. 17–26.

[21] Wiet G, Hittle B, Kerwin T, Stredney D. Translating surgical metrics into automated assessments. Studies in health technology and informatics. 2012;173:543.

[22] Paiva PVF, Machado LS, Valença AMG, Moraes RM. Collaborative Evaluation of Surgical Team's Training Based on Virtual Reality; 2015. Manuscript.

[23] Santos AD, Machado LS, Moraes RM, Gomes RG. Avaliação baseada em lógica fuzzy para um framework voltado à construção de simuladores baseados em RV. In: Anais do XII Symposium on Virtual and Augmented Reality. Natal: Sociedade Brasileira de Computação; 2010. p. 194–202.

[24] Allen B, Nistor V, Dutson E, Carman G, Lewis C, Faloutsos P. Support vector machines improve the accuracy of evaluation for the performance of laparoscopic training tasks. Surgical endoscopy. 2010;24(1):170–178.

[25] Cortes C, Vapnik V. Support-vector networks. Machine learning. 1995;20(3):273–297.

[26] Fan RE, Chen PH, Lin CJ. Working set selection using second order information for training support vector machines. The Journal of Machine Learning Research. 2005;6:1889–1918.

[27] NG A. Support Vector Machine - Optimization Objective;. Acessado em: 2015-12-16. https://class.coursera.org/ml-003/lecture.

[28] Hofmann T, Schölkopf B, Smola AJ. Kernel methods in machine learning. The annals of statistics. 2008;p. 1171–1220.

[29] NG A. Support Vector Machine - Kernels I;. Acessado em: 2015-12-16. https://class.coursera.org/ml-003/lecture.

[30] Platt J, et al. Fast training of support vector machines using sequential minimal optimization. Advances in kernel methods - support vector learning. 1999;3.

[31] Vert JP, Tsuda K, Schölkopf B. A primer on kernel methods. Kernel Methods in Computational Biology. 2004;p. 35–70.

[32] Abakar KA, Yua C. Performance of SVM based on PUK kernel in comparison to SVM based on RBF kernel in prediction of yarn tenacity. Indian Journal of Fibre & Textile Research. 2014;39:55–59.