Uso de funções matemáticas para implementação de ferramentas gráficas em aplicações médicas móveis

Jonathan Alves[1], Edgar Marçal[1]

[1]Instituto Universidade Virtual (UFC Virtual)
Universidade Federal do Ceará (UFC) – Fortaleza, CE – Brasil

alves_aj@windowslive.com, edgar@virtual.ufc.br

Abstract. *This study describes the use of basic mathematical functions in conjunction with the Android API of 2D drawings, which allows the generation of primitive geometric forms. The context of the study of these resources is focused on the health area, in the creation of mobile applications that streamline the processes of physicians with respect to the calculations and plans of their minimally invasive actions. In this study, the Canvas class was used, responsible for the calls of the drawing methods in Views of the Android.*

Resumo. *Este estudo descreve a utilização de funções matemáticas básicas em conjunto com a API do Android de desenhos 2D, que permite a geração de formas geométricas primitivas. O contexto do estudo desses recursos é voltado para a área da saúde, na criação de aplicativos mobile que agilizem os processos dos médicos no que se refere aos cálculos e planejamentos das suas ações minimamente invasivas. Utilizou-se nesse estudo a classe Canvas, responsável pelas chamadas dos métodos de desenhos em Views do Android.*

1. Justificativa e objetivos

Os dispositivos móveis (como celulares e *tablets*) tem se destacado por proporcionar benefícios às atividades diárias dos profissionais de saúde. Esse paradigma, conhecido como *Mobile Health* ou *mHealth*, tem potencial para reduzir os custos na área e melhorar as pesquisas em saúde [Kumar 2013].

Durante o desenvolvimento de aplicações médicas móveis, ou seja, comunicações *mobile* juntamente com tecnologias em rede para sistemas de saúde [Istepanian 2006], um dos problemas que deve ser analisado, em relação aos profissionais da saúde, está na realização de cálculos e medições matemáticas.

Resolveu-se estudar fórmulas matemáticas e inseri-las em métodos que utilizam a *API* (*Application Programming Interface*) de desenhos 2D do sistema operacional *Android*, que contém a classe *Canvas*, com o objetivo de criar uma estrutura lógica gráfica para auxílio de processos médicos.

A classe *Canvas* da *API* de desenhos 2D do *Android* permite construir formas primitivas, tais como, pontos, retas, quadrados, círculos, polígonos complexos, dentre outros. Com esses recursos em mãos é possível, por exemplo, criar uma ferramenta para

auxiliar em planejamentos de procedimentos cirúrgicos minimamente invasivos, ou seja, procedimentos que não se utilizam de cortes profundos e um longo período de internação do paciente.

Dentre as funções matemáticas estudadas as consideradas mais comuns e que podem auxiliar a preparação de uma aplicação móvel para área médica são: a equação reduzida da reta, que se utiliza do coeficiente angular e linear; a equação da perpendicular, que necessita do negativo do recíproco do coeficiente angular; e a equação do ponto médio. Com essas fórmulas básicas é possível iniciar uma aplicação móvel que auxilie no planejamento de uma cirurgia.

2. Metodologia

A metodologia para o desenvolvimento deste trabalho pode ser dividida em três partes. A primeira se relaciona com o estudo da programação da *API* de desenhos 2D do *Android*. A segunda parte é relacionada às funções matemáticas e suas aplicações. Por fim, a terceira parte se estende à utilização da *API* juntamente com as funções matemáticas codificadas na linguagem *Java*.

2.1. API de desenhos 2D do *Android* e sua classe *Canvas*

Para criar desenhos no *Android* é necessário entender 3 classes iniciais fornecidas dentro do pacote *android.graphics.*

A primeira classe é a *Bitmap*, ela permite criar uma área de pixels para geração de desenhos. Quando se trabalha com a criação de novos componentes no *Android* deve-se definir uma classe que estenda de *View*, pois todos os componentes visuais do *Android* são *Views*, logo, não se faz necessário definir um *Bitmap* para trabalhar com o *Canvas*, pois ele tomará toda a *View* por padrão. O *Bitmap* só será usado se for necessário criar um novo *Canvas* ao invés da *View* [Android Developers 2016].

A segunda classe é a *Paint*, utilizada para definir os estilos das formas geométricas. Seus métodos de definição mais comuns são ***setColor()*** (cores de fundo), ***setStrokeWidth()*** (espessura do contorno), ***setTextSize()*** (tamanho do texto, pois a classe *Canvas* também desenha textos), dentre outros.

A terceira classe é a própria *Canvas*, ela será a responsável por fazer as chamadas dos métodos de desenho e escrevê-los na área do *Bitmap* ou *View*. Seus métodos mais comuns e que dispensam definições são ***drawPoint()***, ***drawLine()***, ***drawCircle()***, ***drawRect()***, dentre outros.

2.2. Funções matemáticas utilizadas na área médica

Um dos estudos matemáticos que mais interessa para a área da saúde em relação a análise de exames e planejamento cirúrgico é a geometria analítica, que é estudada a partir da noção de coordenadas e primitivas no sistema cartesiano. Umas das áreas médicas atuais na qual utiliza-se muito a matemática é a Radiologia. Nesta área ter uma

compreensão de matemática é algo fundamental para alcançar um resultado preciso para os pacientes [Delabra 2008].

Faz-se necessário para o estudo das coordenadas a equação da reta, que descreve as propriedades de uma reta (espaço retilíneo entre dois pontos). **Na Figura 1** há a equação reduzida da reta na qual *m* é o coeficiente angular, *b* é o coeficiente linear e *x* é a variável independente que denota o *eixo x* do ponto.

$$y = mx + b$$

Figura 1. Equação reduzida da reta

Para encontrar a reta perpendicular basta calcular o negativo do recíproco do coeficiente angular. No caso, faz-se necessário inverter os números (denominador e numerador) e mudar o sinal do valor do coeficiente angular.

Pode-se também querer encontrar o ponto médio de uma reta, para isso é utilizado as equações da **Figura 2**. Faz-se necessário ter os dois pontos da reta.

$$x = (x1 + x2)/2 \qquad y = (y1 + y2)/2$$

Figura 2. Equação do ponto médio

2.3. Codificação das funções matemáticas em Java no Canvas

Nas **Figuras 3 à 7** pode-se observar com mais detalhes o código fonte em *Java*. O código descreve os métodos *Java* em conjunto com as funções matemáticas.

```
public float getCoeficienteLinear(Ponto ponto1, Ponto ponto2){
    // Retorna o coeficiente angular com o negativo do reciproco para perpendicular
    float coeficienteAngular = negativoReciproco(getCoeficienteAngular(ponto1, ponto2));
    // Coeficiente linear: b = y - mx;
    return (ponto2.getY() - (coeficienteAngular * ponto2.getX()));
}
```

Figura 3. Método *getCoeficienteLinear()* retorna o coeficiente linear de uma reta já criada, para isto é necessário reordenar a equação reduzida da reta. Nota-se que o coeficiente angular utilizado já foi calculado previamente pelo método da Figura 4 e modificado para seu negativo do recíproco da Figura 5.

```
public float getCoeficienteAngular(Ponto ponto1, Ponto ponto2){
    return ((ponto2.getY() - ponto1.getY()) / (ponto2.getX() - ponto1.getX()));
}
```

Figura 4. Retorna o coeficiente angular de uma reta já criada.

```
public float negativoReciproco(Float coeficiente){
    return (-1/(coeficiente));
}
```

Figura 5. Retorna o negativo do recíproco do coeficiente dado.

```
public float getYPerpendicular(Ponto ponto1, Ponto ponto2, float afastamento){
    float coeficienteLinear = getCoeficienteLinear(ponto1, ponto2);
    float coeficienteAngular = negativoReciproco(getCoeficienteAngular(ponto1, ponto2));
    // Equação reduzida da reta y = mx + b
    return ((coeficienteAngular * (ponto2.getX() + afastamento)) + coeficienteLinear);
}
```

Figura 6. Método retorna o *eixo y* de uma reta perpendicular utilizando a equação reduzida da reta. Tem dois parâmetros que representam os pontos de uma reta já formada. A variável *x* da equação pode ser qualquer outro número, já que ela é independente.

```
public void desenhaReta(Canvas canvas, Reta reta, Paint paintLine){
    paintLine.setColor(Color.WHITE);
    paintLine.setStrokeWidth(2);
    canvas.drawLine((float) reta.getxInicio(), (float) reta.getyInicio(),
            (float) reta.getxFinal(), (float) reta.getyFinal(), paintLine);
}
```

Figura 7. Pode-se ver a utilização de uma instância do Canvas, utilizando seu método *drawLine()* que desenha uma linha. O método recebe 5 parâmetros, os eixos *x1, x2, y1* e *y2* de uma reta, e também um *Paint* para estilização.

3. Conclusão

Este estudo se propôs a descrever o uso de tecnologias móveis para agilizar os processos médicos em suas atuações minimamente invasivas. Pode-se observar que muitas vezes faz-se necessário a utilização de formulações matemáticas, que demandam tempo extra de raciocínio do médico para resoluções.

Por exemplo, caso um médico-cirurgião precise saber se o joelho de um paciente é varo ou valgo (deformado para dentro ou para fora, respectivamente), é possível aferir com funções matemáticas. Para isto faz-se necessário descobrir pontos de interesse em uma radiografia da perna, criar retas a partir desses pontos e calcular o ponto médio dessas retas como exposto na **Figura 2**.

A implementação de aplicações que podem fazer cálculos em radiografias, por exemplo, é de grande ajuda para a área da saúde, trazendo também uma melhoria significativa na análise de exames, que, por sua vez, seriam mais ágeis e mais confiáveis no âmbito matemático.

Referências

Android Developers. (2016). *Canvas and Drawables*. Disponível em:<https://developer.android.com/guide/topics/graphics/2d-graphics.html>. Acesso em: 25 de nov. 2016

Delabra, Rebecca. (2008). *Mathematics at work: Health Care*. Achieve. Disponível em: <http://achieve.org/files/MathAtWork-Health.pdf>, p.7. Acesso em: 26 de nov. 2016.

Kumar, Santosh, et al. (2013). *Mobile health technology evaluation: the mHealth evidence workshop*. American journal of preventive medicine. v. 45, n. 2, p. 228-236.

Istepanian, Robert, et al. (2006). *M-Health: Emerging Mobile Health Systems*. Springer Science+Business Media, Inc. p. 23-24.