Um Sistema para Controle de Qualidade de Medicamentos

Maysa Malfiza Garcia de Macedo[1], Aura Conci[1]

[1]Instituto de Computação – Universidade Federal Fluminense (UFF)
Rua Passo da Pátria, 156 - Bloco E - 3° andar. CEP 24.210-240 – Niterói – RJ – Brasil

mmacedo@ic.uff.br, aconci@ic.uff.br

Abstract. *This paper presents a system developed in order to detect fails in the industrial production of pills. The main idea is based on the Hough transform methodology for quality control of drugs in circle form of predefined dimension and perfect shape. This detection occurs during the medicine industrial production to detect pills with imperfect edges.*

Resumo. *Este artigo apresenta um sistema desenvolvido para detectar falhas na produção industrial de comprimidos. A idéia fundamental baseia-se na ampliação da metodologia da transformada de Hough e possibilita detectar se estes estão na forma de círculos predefinidos e perfeitos. Essa detecção ocorre durante a produção de comprimidos com o propósito de detectar principalmente comprimidos cujas bordas apresentem falhas.*

1. Introdução

Hoje, indústrias procuram melhorar sua produção, aumentando a qualidade de seus produtos, diminuindo assim prejuízos e aumentando lucros. Dessa maneira, a pesquisa com base em controle de qualidade é questão fundamental para o provimento de regularidade e uniformidade de um produto. A indústria farmacêutica é uma das que preza pela qualidade de seus medicamentos, pois um comprimido a ser enviado ao mercado, não pode estar com tamanho fora do padrão, imperfeito ou danificado. Uma forma de automatizar o processo de seleção de comprimidos aptos a irem para o mercado, é utilizar um sistema de captura e processamento de imagem, a fim de encontrar possíveis defeitos no produto.

Neste artigo, a teoria de detecção de uma forma presente na imagem é ampliada para possibilitar a detecção de falhas de diversos objetos capturados em uma mesma imagem de modo a ser aplicada no controle de qualidade da produção de pílulas. Algoritmos desenvolvidos para possibilitar o controle de qualidade em comprimidos de forma circular são apresentados. Para ilustrar a eficiência do sistema desenvolvido, exemplos de casos reais são descritos e comentados.

2. Detecção de Círculos e Arcos de Círculos

A Transformada de Hough é uma técnica para reconhecimento, em imagens computacionais, de uma forma que seja facilmente parametrizada, ou seja, que possua uma equação conhecida, tal como reta, círculo ou elipse [Hough 1962]. A idéia básica dessa técnica é transformar a imagem do espaço **x,y** para uma representação na forma dos parâmetros descritos pela curva que se deseja encontrar na imagem. Esta

transformação é aplicada de tal modo que todos os pontos pertencentes a uma mesma curva são mapeados em um único ponto no espaço dos parâmetros da curva procurada.

Para isto, este espaço dos parâmetros é discretizado e representado na forma de uma matriz de inteiros, onde cada posição da matriz corresponde a um intervalo no espaço dos parâmetros. Cada ponto da imagem que satisfizer a equação da forma paramétrica procurada incrementa de uma unidade o contador correspondente a sua posição na representação discretizada. O contador que contiver, no final do processo, os mais altos valores, corresponde ao parâmetro da curva descrita na imagem, permitindo assim sua identificação.

A idéia da transformada de Hough para detecção de círculos [Duda 1972] precisa utilizar uma matriz acumuladora de três dimensões, para que aconteça o armazenamento de suas características básicas: como as coordenadas cartesianas do centro (**a** , **b)** e o próprio raio , **r**, do círculo a ser detectado [Kimme 1975]. Para a aplicação desta transformada, geralmente é realizado um pré-processamento na imagem com o objetivo de identificar as bordas dos elementos que a compõem, de modo que a imagem inicial contenha apenas os contornos das formas existentes na imagem [Watt 1998]. No caso deste trabalho as imagens foram adquiridas através do uso de uma máquina digital e em seguida transformadas para escala de cinza, também foi aplicado um filtro de contorno 3x3 como segue abaixo:

-1	-1	-1
-1	9	-1
-1	-1	-1

Com a fórmula implícita do círculo $(x-a)^2 + (y-b)^2=r^2$, é difícil evidenciar **a** e **b** em termos das demais variáveis, sendo mais adequado utilizar sua forma polar.

$$a = x - r\cos\theta$$

$$b = y - r\sin\theta$$

Para cada ponto da imagem (**x , y**) tem-se 3 valores desconhecidos, a forma de solucionar o problema consiste em fixar um intervalo de raios e para cada raio calcular as coordenadas cartesianas do centro do círculo. Cada centro encontrado associado a um raio, é incrementado de uma unidade na matriz acumuladora. A célula da matriz acumuladora que contém o maior valor, possui índices com as características do círculo a ser detectado. Assim detectam-se os parâmetros, **a**, **b** e **r** da forma presente na imagem.

Para tornar o problema tratável, é necessário estabelecer um intervalo limite para todos os três parâmetros. A coordenada **a** deve ser definida em um intervalo limite de **0** à N_i (coordenada horizontal máxima da imagem). A coordenada **b** define-se em um intervalo limite de **0** à N_j (coordenada vertical máxima da imagem). O raio possui tamanho dentro de limites pré-definidos pelo usuário do programa de detecção de círculos de acordo com as formas que são esperadas na imagem (no caso de comprimidos seus tamanhos esperados). Poderá ser buscado apenas um valor de raio na

imagem (figura 1 à esquerda) ou um intervalo de raios (figura 1 à direita), onde serão detectados os círculos com mais votos dentro desse intervalo. O Algoritmo para detecção de círculos utilizando a transformada de Hough e a discretização descrita acima está representado pela figura 2.

Neste trabalho esta técnica é ampliada para: (1) permitir a identificação na imagem capturada não de uma forma, mas de diversas formas que satisfaçam a equação predefinida; (2) permitir verificar variações de determinado parâmetro da equação buscada; e (3) verificar se as formas buscadas estão completas ou perfeitas. Estas modificações viabilizam a identificação de falhas por imagens em uma indústria automatizada de comprimidos como mostrados nos exemplos da próxima seção. Essa idéia pode ainda ser utilizada para outras formas padrão que usem equações parametrizáveis.

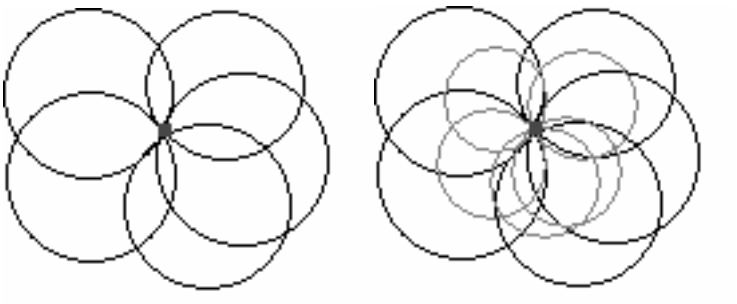

Figura 1. O ponto no centro da figura representa um ponto aceso na imagem para o qual são simulados todos os círculos que possam passar por ele com um determinado raio (esquerda) ou com dois tamanhos diferentes de raio (direita).

circulo_hough()

1- Guardar todos os pixels num array de uma dimensão
2- Limpar acumulador **M(a,b,r)**
3- Para pixel I**(x,y**), se este pixel I**(x,y**) estiver aceso na imagem, execute os comandos de 3 a 11
4- Para **r**>mínimo até **r**<máximo, execute os comandos de 4 a 10
5- Para **θ=0** até **θ=π/2**, execute os comandos de 6 a 9
6- **a= x- r * cos(θ)**
7- **b= y- r * sen(θ)**
8- **M(a,b,r)**=**M(a,b,r)**+1
9- Fim do para os ângulos de **0** à **π/2**
10- Fim do para todos os raios de mínimo ao máximo
11- Fim do para todos os pixels da imagem

Figura 2. Algoritmo de extração de características da imagem e armazenamento em uma matriz acumuladora.

De acordo com a necessidade de detecção de um determinado tipo de círculo, no algoritmo **circulo_hough()**, pode ser utilizado um único raio ou um intervalo de raios.

Para cada tamanho de raio, são calculados possíveis coordenadas de centro, para ângulos que variam de **0** à **2π** radiano.Caso o círculo detectado não esteja perfeitamente sobre o círculo original, é necessário submeter o resultado a uma checagem de sua vizinhança, o que é feito através do algoritmo **busca_aceso( )** descrito na figura 3. A idéia básica deste algoritmo é verificar se a vizinhança da imagem original contém um pixel aceso, adotando o mesmo como parte do círculo identificado. O que funciona como uma taxa de tolerância para o padrão presente na imagem em análise visando a detecção de possíveis elementos.

busca_aceso()

1- Variável **ap** recebe o tamanho da vizinhança, **i** e **j** recebem as coordenadas do pixel de onde será calculada a vizinhança
2- Para **n**= **-ap** até **n**= **ap**, execute os comandos de 3 a 7
3- Para **m**= **-ap** até **m**= **ap**, execute os comandos de 4 a 6
4- Se o pixel **I(i+n , j+m)** estiver aceso, então execute o comando 5
5- **busca_aceso=true,** sai do procedimento
6- Faça laço 3-6 até **m=ap**
7- Faça laço 2-7 até **n=ap**
8- **busca_aceso=false**

Figura 3. Algoritmo para buscar pixel aceso numa vizinhança estabelecida.

Depois se deve voltar ao espaço da imagem para nesta identificar as formas detectadas, o que é feito no algoritmo **circulo_inversa( )** mostrado na figura 4. Antes de executar este algoritmo é feito uma busca dos maiores valores na matriz acumuladora. O número de valores dependerá da quantidade de círculos a serem identificados. Neste algoritmo a variável booleana **dentro** terá valor **false** se o pixel em questão não estiver no arco de círculo e terá valor **true** se o pixel estiver no arco de círculo. Caso o círculo seja completo e sem falhas como na figura 7 (à esquerda), então os pontos de início e fim de arco de círculo terão valor igual a zero. Caso o mesmo contenha falhas ou trate-se de um arco de círculo, como na figura 7 (à direita), os pontos de início e fim terão valor diferente de zero.

Os algoritmos desta seção foram implementados em um programa para detecção de círculos e arcos de círculos. O programa permite também a detecção de imagens com falhas, como mostrado nos exemplos da próxima seção.

circulo_inversa()

1- Inicializar **x_menor**, **y_menor**,**x_maior**,**y_maior**,**anterior_x**, **anterior_y**
2- Para cada célula na matriz acumuladora **M(a,b,r**), execute os comandos de 3 a 27
3- Para **θ =0** até **θ = 2π**, execute os comandos de 4 a 23
4- **x= a + r * cos(θ)**
5- **y= b + r * sen(θ)**
6- Se (**x>0**) e (**x<Ni**) e (**y>0**)e (**y<Nj**), ou seja, se estiver no domínio da imagem, execute os comandos de 7 a 23
7- Se o pixel **I(x,y)** estiver aceso na imagem original, execute os comandos de 8 a 12
8- Se a variável **dentro** for igual a **false**, significa que o pixel não está sobre o círculo, então execute os comandos de 9 e 10
9- **x_menor** = **x**, **y_menor** = **y**
10- **dentro**=true
11- O **pixel IS(x,y)** na imagem de saída é aceso
12- Senão, se o pixel **I(x,y)** não estiver aceso na imagem original, execute os comandos de 13 a 22
13- **aceso**=busca_aceso()
14- Se **aceso** for igual a **true**, então execute os comandos de 15 a 18
15- Se a variável **dentro** for igual a **false**, então execute os comandos de 16 e 17
16- **x_menor** = **x**, **y_menor** = y
17- **dentro**=true
18- O **pixel IS(x,y)** na imagem de saída é aceso
19- Senão, se **aceso** for igual a **false** e a variável dentro for igual a **true**, então execute de 20 a 21
20- **x_maior** = **anterior_x**, **y_maior** = **anterior_y**
21- **dentro**=false
22- **anterior_x=x**, **anterior_y=y**
23- Faça o laço 3-23 para todos os ângulos de **0** a **2π**
24- Para cada circulo identificado, execute o comando 26
25- Armazenar os pontos de um arco de círculo **(x_menor,y_menor)** e **(x_maior,y_maior)**
26- Faça o laço 24-26 para todos os círculos identificados
27- Faça o laço 2-27 para todos as células da matriz acumuladora.

Figura 4. Algoritmo para extração dos parâmetros dos círculos a partir de uma matriz acumuladora.

3. Testes de funcionalidade do sistema

A primeira imagem testada é a mostrada na figura 5 de 249x175 pixels. O círculo detectado é mostrado à direita nesta imagem, onde a posição do centro com coordenadas (95,102) e o círculo localizado são detectados e mostrados sobre a imagem inicial. Foram usados raio=69, com tolerância de erro de **2** pixels e intervalos para a matriz acumuladora de **1** pixels para os parâmetros **a** e **b.**

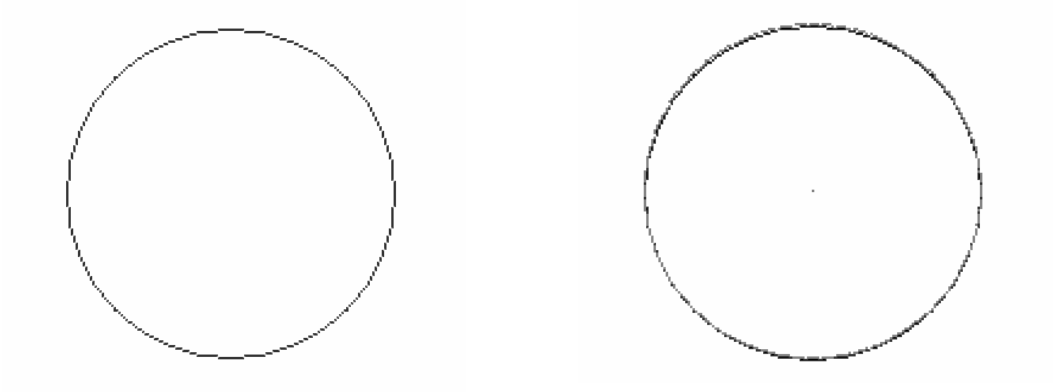

Figura 5. Exemplo 1: um círculo completo: imagem inicial (à esquerda) e resultado obtido (à direita) na cor cinza.

O segundo exemplo demonstra a viabilidade de detecção de um arco de círculo. A imagem testada é a mostrada na figura 6. Foi usado para a matriz acumuladora discretização de um em um pixel para **a** e **b**. E 1 pixel de tolerância para possíveis deslocamentos dos arcos. Um raio de 54 pixels é considerado. O resultado obtido é mostrado na imagem à direita da figura 6, com coordenadas de centro (62,66) e pontos inicial e final do arco de (116,67) e (8,67) respectivamente.

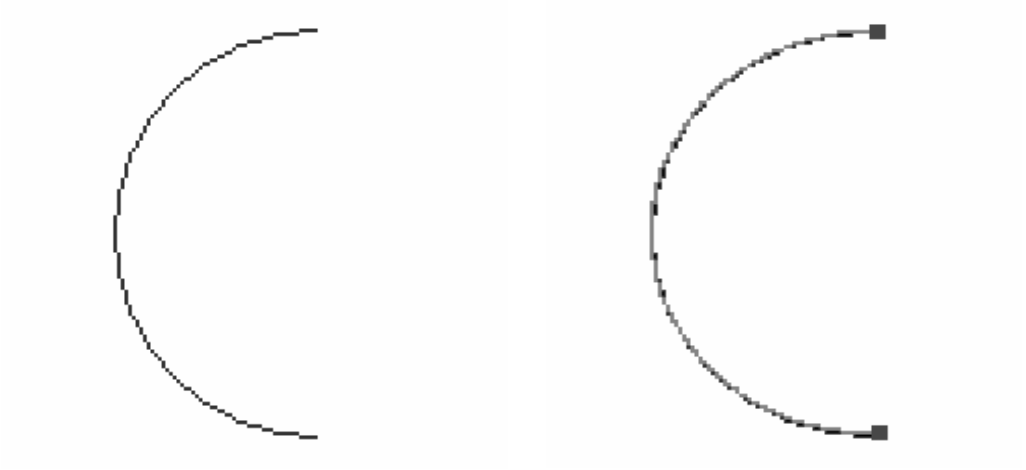

Figura 6. Exemplo 2: arco de círculo: imagem inicial de dimensão 192x 140 (à esquerda) e elementos detectados (à direita).

O terceiro exemplo considera a detecção de três círculos completos (à esquerda na figura 7). Foi aplicado um intervalo para raio de 11 a 27 pixels, intervalos para a matriz acumuladora de **1** pixels para os parâmetros **a** e **b** e tolerância de 2 pixels para possíveis deslocamentos**.** Foi detectado um círculo com centro (107,129) e raio de 27

pixels, um outro círculo com centro (39,37) e raio de 23 pixels e um último círculo com centro (24,144) e raio de 11 pixels.

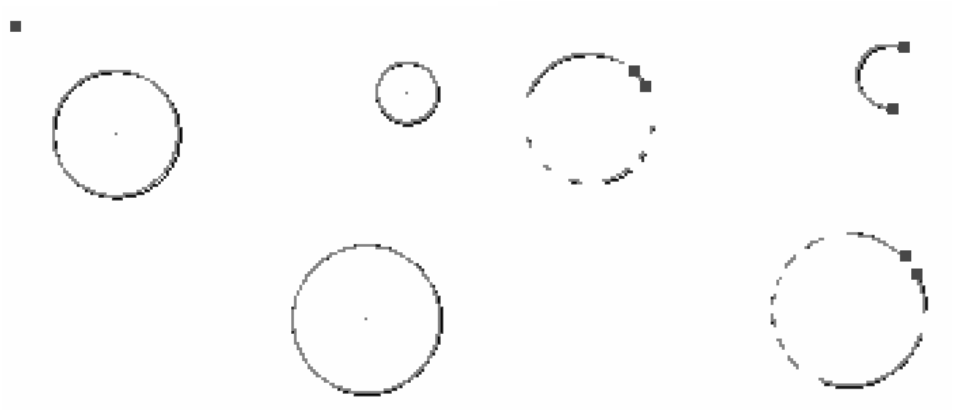

Figura 7. Exemplo 3: (à esquerda) três círculos completos detectados. Exemplo 4: (à direita) um arco de círculo e dois círculos com falhas detectadas.

E o quarto exemplo considera os círculos com falhas parciais mostrados na imagem à direita da figura 7. Foi aplicado o mesmo intervalo para raio do exemplo 3, mesma discretização da matriz acumuladora, e mesma tolerância a deslocamentos. Foi detectado um círculo com centro (107,129), raio de 27 pixels e pontos sob o círculo de coordenadas (94,153) e (90,150), um segundo círculo com centro (39,37), raio de 23 pixels e pontos sob o círculo com coordenadas (23,53) e (27,57), e um terceiro círculo com centro (24,144), raio de 11 pixels e pontos inicial e final do arco com coordenadas (33,149) e (15,149). Através destes quatro primeiros testes, conclui-se que as detecções de círculos ocorreram com sucesso, tanto com imagens de círculos completos quanto com círculos com falha ou com arcos de círculo.

A segunda série de testes é dedicada a imagens reais de comprimidos e buscam verificar se há comprimidos danificados. No caso de falhas, os pontos de início e fim estarão sob o comprimido. As figuras 8 e 10 (à esquerda) mostram as imagens originais, as figuras 8 e 10 à direita mostram as imagens pré-processadas, apenas contendo as bordas dos comprimidos, e as figuras 9 e 11 são as imagens detectadas. As imagens originais nestes casos possuem dimensões 192x153 pixels (figura 8) e 220x171 pixels (figura 10). Nos dois exemplos, foram utilizados raios de 15 pixels e intervalos para a matriz acumuladora de **1** pixel para os parâmetros **a** e **b** . Para o exemplo 5 foi dada uma tolerância de 1 pixel para possíveis deslocamentos e para o exemplo 6, tolerância de 3 pixels**.** O procedimento mostrado a cima pode ser realizado a cada período de tempo, enviando um sinal ao encontrar irregularidade no medicamento, sincronizando a captura com a velocidade da produção ou da correia transportadora da indústria.

Todos os testes foram realizados em uma máquina AMD Athlon 2000 com 1.7 Ghz, 256 Mb de memória e 40Gb de disco. A tabela 1 mostra o tempo gasto pelo sistema na detecção de cada um dos exemplos descritos.

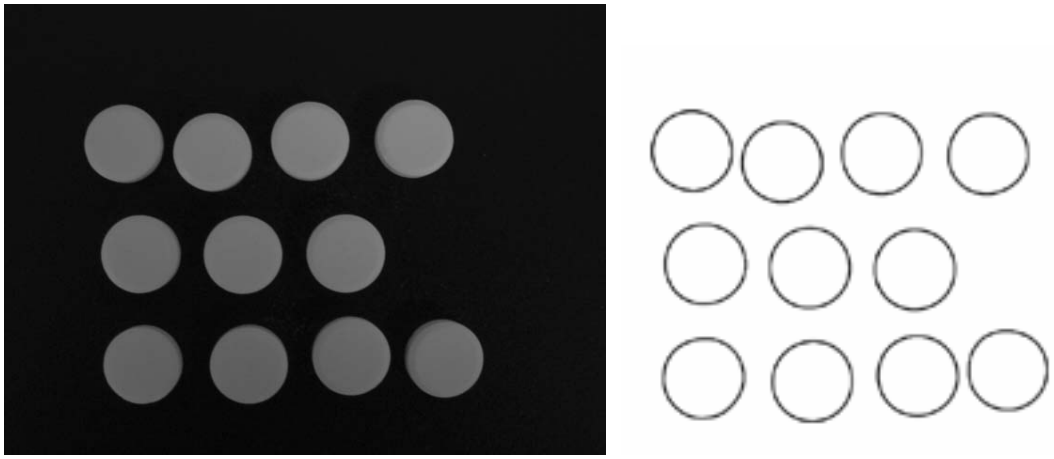

Figura 8. Imagem original de comprimidos à esquerda e à direita imagem pré-processada, mostrando apenas as bordas dos comprimidos.

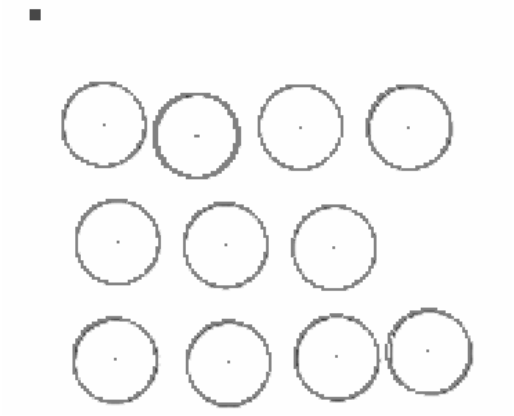

Figura 9. Exemplo 5: Imagem detectada. 11 círculos identificados. O ponto no canto superior esquerdo mostra que não há falhas nos comprimidos.

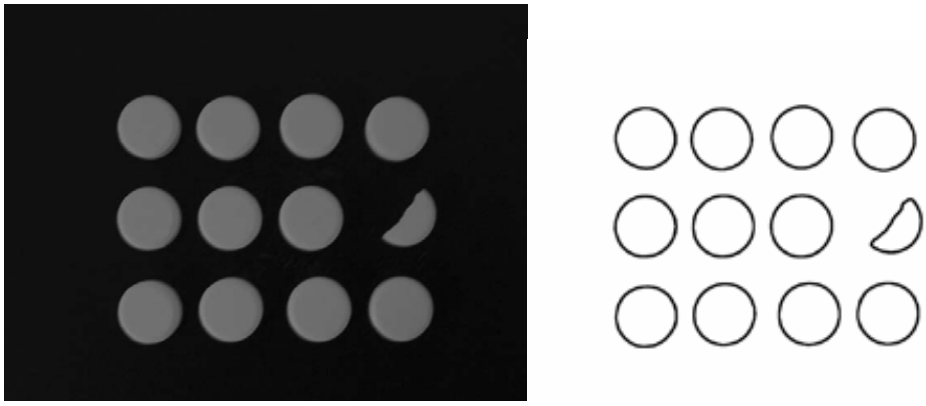

Figura 10. Imagem original de comprimidos à esquerda e à direita imagem pré-processada, mostrando apenas as bordas dos comprimidos.

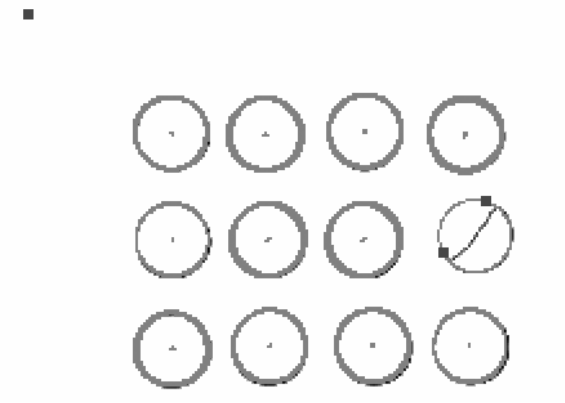

Figura 11. Exemplo 6:Imagem detectada. Os pontos sobre um dos círculos, indicam que o mesmo está danificado. A permanência do ponto no canto superior esquerdo significa que existe pelo menos um comprimido em perfeito estado.

Tabela 1. Tempo de execução de cada teste

Figura	Dimensão	Tempo de Execução(segundos)
exemplo 1	249x175	0,094
exemplo 2	192x140	0,047
exemplo 3	192x140	1,172
exemplo 4	192x140	0,718
exemplo 5	192x153	0,188
exemplo 6	222x171	0,219

4. Conclusão

Neste trabalho foi apresentado um sistema que permite a detecção de formas circulares completas ou não, com ou sem falhas, e verificar a existência ou não destas falhas. Como exemplo aplica-se o método a diversas imagens sintéticas e reais de comprimidos no sentido de mostrar sua eficiência. A idéia aqui apresentada para formas circulares pode ser estendida para detecção de outras formas equacionáveis. No caso específico da indústria farmacêutica seria principalmente útil sua extensão para formas elípticas [XIE, 2002]. O sistema se mostrou viável e de implementação simples e eficiente.

Referências

Duda, R. , Hart, P. (1972) “Use of Hough transformation to detect lines and curves in pictures”, In: Comunications of ACM, Volume 15, Number 1.

Hough, P. (1962) "Method and means for recognizing complex patterns ", In U.S Patent 3 069 654, December."

Kimme, C. , Ballard, D. and Sklansky, J. (1975) “Finding circles by an array of accumulators”, In: Comunications of ACM, Volume 18, Number 2.

Watt, A. C. , Policarpo, A. (1998) The Computer Image, Addison-Wesley Pub Co (Net).

Xie, Y. , Ji, Q. (2002) “A New Efficient Ellipse Detection Method”, In: IEEE, Volume 2.