Segunda lista de exercícios: Visão Computacional
Respostas Teóricas

Questão 1:

- Detector de Harris

O detector de bordas Harris é um operador de detecção de bordas que é comumente usado em algoritmos de visão computacional para extrair cantos e inferir características de uma imagem. Os passos realizados pelo algoritmo são descritos abaixo:

1. Pegue a escala de cinza da imagem original
2. Aplique um filtro gaussiano para suavizar qualquer ruído
3. Aplique o operador Sobel para encontrar os valores de gradiente x e y para cada pixel na imagem em tons de cinza
4. Para cada pixel p na imagem em tons de cinza, considere uma janela 3×3 ao seu redor e calcule a função de força do canto. Chame isso de valor de Harris.
5. Encontre todos os pixels que excedem um determinado limite e são os máximos locais dentro de uma determinada janela (para evitar duplicações redundantes de recursos)
6. Para cada pixel que atende aos critérios em 5, calcule um descritor de característica.

De acordo com a implementação do OpenCV, o detector possui os seguintes parâmetros:

- img : Imagem de entrada. Deve ser em tons de cinza e tipo float32.
- blockSize: É o tamanho da vizinhança considerada para detecção de canto
- ksize: Parâmetro de abertura da derivada de Sobel utilizada.
- k: Parâmetro livre do detector de Harris na equação.

Os parâmetros blockSize e ksize são utilizados em 2, 3 e 4 dos itens listados acima e k é uma constante determinada empiricamente no intervalo [0,04,0,06].

- Detector de Shi-Tomasi

O detector de Shi-Tomasi é baseado no detector de Harris e funciona da mesma forma, contudo produz resultados melhores que o detector de Harris por aprimorar a Função de pontuação de Harris que anteriormente era dada por $R = \lambda1\lambda2 - k (\lambda1 + \lambda2)^2$ e tonou - se $R = min (\lambda1, \lambda2)$ tal modificação proporcionou melhores resultados que o detector de Harris.

De acordo com a implementação do OpenCV, o detector de Shi-Tomasi possuem os seguintes parâmetros:

- img: Insira uma imagem de canal único de 8 bits ou de ponto flutuante de 32 bits. cantos Vetor de saída de cantos detectados.

- maxCorners: Número máximo de cantos a serem retornados. Se houver mais cantos do que os encontrados, o mais forte deles será retornado. maxCorners <= 0 implica que nenhum limite para o máximo é definido e todos os cantos detectados são retornados.

- qualitylevel: Parâmetro que caracteriza a qualidade mínima aceita dos cantos da imagem. O valor do parâmetro é multiplicado pela melhor medida de qualidade de canto, que é o autovalor mínimo ou a resposta da função de Harris. Os cantos com medida de qualidade menor que o produto são rejeitados.
- minDistance: Distância euclidiana mínima possível entre os cantos devolvidos.

- blocksize: Tamanho de um bloco médio para calcular uma matriz de covariação derivada sobre cada vizinhança de pixel.
- useHarrisDetector: Parâmetro que indica se deve usar um detector Harris .
- k: Parâmetro livre do detector Harris.

De posse do resultado da função de pontuação R, temos que todos os cantos abaixo do nível de qualidade são rejeitados. Em seguida, ele classifica os cantos restantes com base na qualidade em ordem decrescente. Então a função pega o primeiro canto mais forte, joga fora todos os cantos próximos no intervalo de distância mínima e retorna N cantos mais fortes.

- Fast Detector

O detector FAST usa um círculo de 16 pixels (um círculo de Bresenham de raio 3) para classificar se um ponto candidato **p** é realmente um ponto de aresta. Cada pixel no círculo é rotulado do número inteiro de 1 a 16 no sentido horário. Se um conjunto de N pixels contíguos no círculo são todos mais brilhantes que a intensidade do pixel candidato **p** (indicado por **Ip**) mais um valor de limiar t ou todos mais escuros que a intensidade do pixel candidato **p** menos o valor de limiar t, então **p** é classificado como canto. É um dos algoritmos de detecção mais rápidos e pode ter seu desempenho aprimorado usando aprendizagem de máquina.

O funcionamento do algoritmo pode ser resumido nas seguintes etapas:

1. Selecione um pixel **p** na imagem que deve ser identificado como um ponto de interesse ou não. Seja sua intensidade EU p.
2. Selecione o valor limite apropriado **t**.
3. Considere um círculo de 16 pixels ao redor do pixel em teste.
4. Agora o pixel p é um canto se existe um conjunto de N pixels contiguos no círculo (de 16 pixels) que são todos mais brilhantes do que I_p+ t, ou tudo mais escuro que I_p− t.
5. Um teste de alta velocidade foi proposto para excluir um grande número de não-cantos

Questão 2:

- **SIFT**

SIFT é um algoritmo que preza pela invariância da escala e rotação da imagem, e tem como pilares de seu funcionamento a localidade, a distinção, a quantidade, Eficiência e Extensibilidade. Este algoritmo pode ser dividido nas etapas abaixo:

- **Seleção de pico de espaço de escala:** local potencial para encontrar recursos.O espaço de escala de uma imagem é uma função L(x,y,σ) que é produzida a partir da convolução de um kernel gaussiano (Blurring) em diferentes escalas com a imagem de entrada.

- **Localização do ponto-chave**: Localizando com precisão os pontos-chave do recurso, tal abordagem é semelhante à usada no detector Harris Corner para remover recursos de borda e é necessário para eliminar os falsos pontos-chave encontrados após o espaço de escala.
- **Atribuição de Orientação:** Atribuir orientação a pontos-chave com a mesma localização e escala, mas direções diferentes. Contribui para a estabilidade da correspondência.
- **Descritor de ponto-chave :** Descrevendo os pontos-chave como um vetor de alta dimensão. Neste ponto, cada ponto-chave tem uma localização, escala, orientação. Em seguida é computado um descritor para a região da imagem local sobre cada ponto chave que seja altamente distintivo e invariável quanto possível a variações como mudanças no ponto de vista e iluminação.
- **Correspondência de pontos-chave**: Os pontos-chave entre duas imagens são combinados identificando seus vizinhos mais próximos. Mas, em alguns casos, a segunda correspondência mais próxima pode estar muito próxima da primeira.

De acordo com a implementação do opencv, os seguintes parâmetros são necessários:

- **nfeatures:** número de melhores features a serem retidas.
- **nOctaveLayers**: número de camadas em cada oitava. 3 é o valor usado no papel D. Lowe. O número de oitavas é calculado automaticamente a partir da resolução da imagem.
- **contrastThreshold:** limiar de contraste usado para filtrar características fracas em regiões semi-uniformes (baixo contraste). Quanto maior o limiar, menos características são produzidas pelo detector.
- **edgeThreshold:** limite usado para filtrar recursos semelhantes a bordas. Observe que o seu significado é diferente do contrasteThreshold, ou seja, quanto maior o edgeThreshold, menos recursos são filtrados (mais recursos são retidos).
- **sigma:** sigma do Gaussiano aplicado à imagem de entrada na oitava #0. Se sua imagem for capturada com uma câmera fraca com lentes macias, convém reduzir o número.
- **descriptorType**: tipo de descritores. Apenas CV_32F e CV_8U são suportados.

● SURF (Speeded Up Robust Features)

SURF é um algoritmo rápido e robusto para representação local, invariante de similaridade e comparação de imagens. O principal interesse da abordagem SURF está em sua rápida computação de operadores usando filtros de caixa, permitindo assim aplicações em tempo real, como rastreamento e reconhecimento de objetos. Ele acontece em duas etapas: extração de recursos e descrição de recursos. Na primeira etapa, o algoritmo utiliza aproximação de matriz Hessiana simples para detecção de pontos de interesse, dependendo do determinante dessa matriz para selecionar localização e escala. Na segunda etapa, é fixada uma orientação ao redor do ponto escolhido e é construída uma região quadrada alinhada à orientação selecionada. Dela é extraído o descritor SURF.

De acordo com a implementação do opencv, os seguintes parâmetros são necessários:

- **hessianThreshold:** limiar para detector de ponto chave hessiano.
- **nOctaves:** número de oitavas da pirâmide que o detector de ponto chave usará.
- **nOctaveLayers:** número de camadas de oitava dentro de cada oitava.

- **extended:** sinalizador de descritor estendido (true - descritores estendidos de 128 elementos; false - descritores de 64 elementos).
- **upright:** sinalizador de recursos para cima para a direita ou girado (verdadeiro - não calcula a orientação dos recursos; falso - calcula a orientação).

● BRIEF (Binary Robust Independent Elementary Features)

O BRIEF é muito rápido tanto para construir quanto para combinar. Supera facilmente outros descritores como SURF e SIFT em termos de velocidade e taxa de reconhecimento em muitos casos. Ele depende de um número relativamente pequeno de testes de diferença de intensidade para representar um patch de imagem como uma string binária. O kernel gaussiano é usado para suavizar a imagem, então um descritor é computado para cada ponto-chave (descrito por um vetor de recursos que é uma string de 128 a 512 bits).

De acordo com a implementação do opencv, os seguintes parâmetros são necessários:

- **bytes**: comprimento do descritor em bytes, os valores válidos são: 16, 32 (padrão) ou 64 .
- **use_orientation**: padrões de amostra usando orientação de pontos-chave, desabilitada por padrão.