A very brief introduction of what is known about vision experimentally

Neurons, neural circuits, and brain regions along the visual pathway

Multiple brain areas are involved in the processing of visual information. Visual inputs from the retina are transmitted to the lateral geniculate nucleus (LGN) of the thalamus and then form projections to the primary visual cortex (V1, striate cortex). Afterwards, information is transmitted towards the frontal areas to extrastriate cortex (V2, V3, and V4, middle temporal area MT/V5) and other brain regions such as inferotemporal cortex (IT), lateral intraparietal area (LIP), and frontal eye field (FEF). These areas can be divided into the ventral visual pathway ("what": recognition; V2, V4, IT) and the dorsal visual pathway ("where": motion, speed; V3, MT, LIP).

Each neuron typically responds to a limited region of the visual space, known as the receptive field. The size of the receptive fields (measured in visual angle) increase along the visual pathway. Visual inputs that excite retinal neurons are dots in contrasting backgrounds. Neurons in V1, on the other hand, are excited by static or moving bars and edges of varying orientations. Further along the visual pathway, receptive fields become more complex and depend on the animal's conscious state and what the animal is paying attention to.

Retina

The signals from the photoreceptors are processed by horizontal, bipolar, and amacrine cells. They are then transmitted to the retinal ganglion cells that project to the central brain through the optic nerve. The response of a typical ganglion cell is approximated as a weighted summation of photoreceptor signals. The weights are determined by a filter $K(x)$ that is non-zero within the receptive field of the ganglion cell. This filter is a center-surround receptive field and is modeled as a difference between two Gaussians:

$$K(x) = \frac{\omega_c}{\sigma_c} exp(-\frac{x^2}{2\sigma_c^2}) - \frac{\omega_s}{\sigma_s} exp(-\frac{x^2}{2\sigma_s^2})$$

Receptive field mapping is done by shining light at different locations of the receptive field (S. W. Kuffler, 1953), by using disks of light centered at the receptive field (D. H. Hubel, T. N. Wiesel, 1959), or by using grating patterns with varying frequency. A grating can be described as a cosine wave with a spatial frequency k, grating amplitude S_k, and spatial phase ϕ:

$$S(x) = S_k\ cos(kx + \phi) + \text{constant}$$

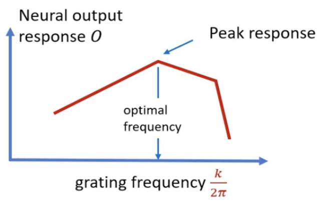


Fig. 1. Contrast sensitivity function of a retinal ganglion cell.

The neural response as a function of the grating frequency k is called the contrast sensitivity function $g(k)$ (Fig. 1). It peaks at the frequency where the wavelength is similar to the size of the receptive field.

The receptive field can be approximated as a weighted summation Fourier waves. The amplitude of the waves scales with the contrast sensitivity function $g_c(k)$. Then, the spatial shape of the receptive field $K(x)$ can be constructed as a weighted summation of cosine waves by measuring the contrast sensitivity function $g_c(k)$ through neuron's responses to perfectly aligned gratings:

$$O = \int dx\ K(x)\ S_k\ cos(kx+\phi) \propto g_c(k)\ S_k\ cos(\phi)$$

The spatial input $S(x)$ and spatial filter $K(x)$ can be generalized to spatiotemporal input $S(x,\ t)$ and spatiotemporal filter $K(x,\ t\ -\ t')$ so that the neural response evolves in time:

$$O(t) = \int dt'\ dx\ K(x,t-t')\ S(x,t') + \text{spontaneous firing rate}$$

The neuronal response right after stimulus onset (small $t > 0$) is called the transient response; otherwise, it is sustained response ($t \to \infty$). The impulse response function describes a neuron's reaction to a brief input signal (Fig. 2).

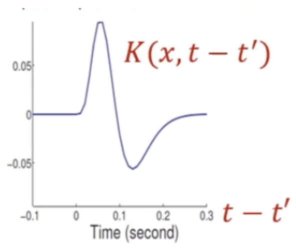


Fig. 2. Impulse response function of a model neuron to a brief input signal. The neuron initially becomes excited; the excitation is followed by a period of inhibition and recovery to the baseline.

The center-surround receptive field model can be expanded to include time as well:

$$K(x,t) = \frac{K_c(t)\ \omega_c}{\sigma_c^2}\ exp(-\frac{|x|^2}{2\sigma_c^2}) - \frac{K_s(t)\ \omega_s}{\sigma_s^2}\ exp(-\frac{|x|^2}{2\sigma_s^2})$$

The filters $K_s(t)$ and $K_c(t)$ have the same shape as the impulse response function (Fig. 4). However, the surround filter $K_s(t)$ has a longer temporal width than $K_c(t)$. Such filters are sensitive to both spatial and temporal contrasts in visual inputs.

The retina has three types of cones: red (long wavelength, L cones), green (medium wavelength, M cones), and blue (show wavelength, S cones) cones. The response of a retinal ganglion cell is the summation of the contributions of different cones. Cones are more densely packed around the fovea and their density reduces with eccentricity. The sizes of the receptive fields increase with eccentricity. Another way to represent the 3-dimensional color inputs is to use luminance, red-green, and blue-yellow channels. The signals in these channels are decorrelated from each other.

The two main types of cells in the retina are the parvocellular (P) cells and the magnocellular (M) cells. The P cells are more common, and have better spatial resolution but worse temporal resolution. M cells display non-linear responses to visual inputs, are more sensitive to luminance inputs and not sensitive to color differences.

Primary visual cortex (V1)

Visual inputs from the retinal ganglion cells travel to V1 through LGN. V1 of different hemispheres receives inputs from half of the visual field. A larger area of V1 is dedicated for processing visual information from the fovea, and the devoted V1 area decreases drastically as the eccentricity increases (cortical magnification factor). The receptive fields in V1 resemble multiple retinal ganglion receptive fields added together, and are tuned to (tilted) bars and edges. They are modeled using Gabor filters:

$$K(x,y) \propto exp(-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2})\ cos(\hat{k}x + \phi)$$

The preferred frequencies are in the Gaussian envelope around the center frequency $\widehat{k}$. The range of the preferred frequencies is inversely related to the σ_x. Cells tuned to different frequency ranges are needed to cover the whole frequency range at a single spatial location (multiscale coding). Sensitivity to chromatic inputs is focused on lower spatial frequencies than sensitivity to luminance. Thus, many color-selective neurons are not tuned to orientation and have larger receptive fields. Spatiotemporal receptive fields are sensitive to both contrast and time, and are able to detect motion. These filters can be space-time separable, and space-time non-separable. A space-time non-separable can be created by combining two or more space-time separable filters and are selective to motion direction.

Binocular disparity refers to the difference in image location of an object seen by the left and right eyes. It signals visual inputs at different depths in a 3-dimensional visual world. Monocular neurons prefer inputs from one eye over another eye (ocular dominance) and binocular neurons are equally sensitive to inputs from both eyes.

V1 neurons belong to one of the two classes: either simple cells, or complex cells. Simple cells have static nonlinearity and can be approximated as linear. They prefer features such as orientation, scale, color, motion direction, disparity, and spatial location of visual inputs. Simple cells can be on-cells and off-cells and detect opposite features. These on- and off-cells are combined to approximate a linear filter. A complex cell receives inputs from two linear simple cells in a quadrature pair (energy model):

$$L_1 = \int dx\ K_1(x)S(x),\ L_2 = \int dx\ K_2(x)S(x) \rightarrow E \equiv L_1^2 + L_2^2$$

A complex cell is sensitive to relevant visual features (e.g., orientation). However, it is insensitive to small changes in spatial location and the contrast polarity (e.g., not sensitive to the phase of the grating).

Suppression to neural responses by visual inputs outside the receptive field is called surround (contextual) suppression (extra-classical receptive fields). This suppression is the strongest when the contextual input has the same vertical orientation as the visual input within the classical receptive field. Some contextual inputs can increase the response (contextual facilitation). Contextual inferences are likely caused by interactions between neighboring V1 neurons and feedback from higher visual areas.

Higher visual cortical areas

V2 follows V1 along the visual pathway and has a similar size. Neural properties are similar to those in V1, however, the receptive fields are slightly larger. V2 neurons are sensitive to real and illusory contours, border ownership, and depth order between surfaces.

MT / V5 is along the dorsal visual pathway. Most neurons are tuned to motion direction and binocular disparity and their receptive field diameter is about 10 times larger than those of V1 neurons. MT responds to plaids (pattern cells) or component motions (component cells), and groups of moving dots.

V4 is in the ventral visual stream. Its neurons are tuned to similar features as V1. The diameter of receptive fields in V4 is 4-7 times larger as compared to V1. V4 lesions mildly impair simple visual perception, however, they severely impair shape discrimination.

IT in the ventral stream. Its neurons have large receptive fields that include the central fovea. They are driven by moderately complex shapes (face, bush, hand, etc.) rather unselectively. However, some patches include cells exclusively selective to face-like features or body parts.

Central vision loss is caused by macular degeneration through loss of retinal ganglion cells and results in difficulties recognizing faces, reading, etc. Tunnel vision is caused by the vision loss in the peripheral visual field (glaucoma, damage to optic nerve) and causes difficulties in navigation. Damage to V1 makes the person clinically blind. However, there are some residual visual abilities called blindsight (deny seeing an object but can navigate around it). Hemineglect is a deficit of attentional awareness to a half of the visual field and is caused by damage to one (right) hemisphere of the brain.

Eye movements and visual attention

Visual acuity is poor away from the center of gaze. As a result, the gaze shifts around 3 times per second. Gaze trajectories are affected by the internal goal of the observer.

Two types of eye movements include saccades (jump between objects) and smooth pursuits (following an object). Eye movements are related to attention. For example, just before an impeding saccade, object discrimination is best at the destination of the saccade as the attention is directed to that location. It is possible to keep the gaze fixed while directing attention elsewhere; however, shifting gaze without shifting attention is impossible.

Multiple brain regions are involved in eye movements. SC receives inputs from the retina and visual cortical areas and sends commands to control saccadic eye movements. The frontal eye field can bypass SC to directly command eye movements. The upper layers of SC (sensory) have a retinotopic map of the visual field with small receptive fields. In the deeper layers of SC (motor) the neurons are active before or during saccades to their movement fields (the movement fields coincide with the receptive fields in the upper layers).

Most LIP and FEF neurons have visual receptive fields and many respond before or during saccades to their movement fields. Electrical stimulation of V1, V2, superior colliculus, LIP, and FEF can produce saccadic eye movements towards the receptive field or the movement field of the stimulated neurons. FEF stimulation still evokes saccadic eye movements if the superior colliculus is lesioned (has direct command lines to the brain stem). Lesioning SC reduces the number of spontaneous saccades and short latency saccades (express saccades) are eliminated. FEF lesions eliminate memory-guided or non-visually guided saccades (unable to move eyes in response to motor commands). V1 lesions abolish all visually-guided saccades until 2 months after the lesion which means that direct visual inputs from the retina to SC are not enough to drive saccades in monkeys.

Neurons within LIP, FEF, and SC increase their responses to visual input within their receptive fields if this input is the target of an upcoming saccade. Moreover, they shift their visual receptive fields so that they respond to visual inputs that are about to be brought into the classical receptive fields by the impending saccades, a process known as receptive field remapping.

Paying attention to a location is similar to stimulating the corresponding SC neuron. Visual discrimination is better at the receptive field of the more active neuron in LIP. Subthreshold stimulation of FEF or SC improves task performance inside the movement fields of the stimulated neurons.

Attention is measured by its effect to increase the performance or speed of object recognition at the attended location versus a non-attended location or feature. Attention can be controlled endogenously (voluntarily) or exogenously (involuntarily). Usually, the effects of endogenous attention are measured. They are observed in V2, V4, IT, MT; the effects in V1 are very weak. Attention increases sensitivity and effective input strength, scales feature tuning curve and changes feature tuning width (biased competition: visual inputs compete for being represented in the brain).

Exogenous attentional shift is caused by external visual inputs, so it is difficult to distinguish between (1) changes of external visual inputs, and (2) changes in exogenous attention as the reason for changes in the neural responses. Although top-down attentional influences on V1 responses are weak or absent, contextual influences in V1 could depend on what task a monkey is performing and on the monkey's experience.

Visual behavior

A typical visual behavioral experiment involves an observer performing a visual task and their reaction is observed by pressing buttons, tracking gaze position or pupil size, measuring their brain waves. Categories of visual behavior can be viewed in terms of encoding (visual input samping and image processing), selection (visual attention and segmentation), and decoding (recognition).

Examples of remarkable visual behavioral phenomena include: inattentional blindness, random dot stereograms, visual crowding, Adelson’s checker-shadow illusion, Kanizsa’s triangle, hyperacuity, reverse Phi motion illusion, scintillating grid illusion, occlusion and recognition, ambiguous perception, binocular rivalry, gaze captured by a non-distinctive and task-irrelevant object.

Questions

1. Electrical stimulation of V1, V2, superior colliculus, LIP, and FEF produces saccadic eye movements towards the receptive / movement field of the stimulated neurons. Which brain region stimulation still evokes saccadic eye movements if the superior colliculus is lesioned?
    a. V1
    b. V2
    c. LIP
    d. FEF

    Explanation: Stimulation of the neurons in the frontal eye field (FEF) evokes saccadic eye movements if the superior colliculus is lesioned because FEF has direct connections to both SC and the brainstem.

2. What visual phenomenon is characterized by the brain's inability to consciously perceive an object despite it being in the visual field?
    a. Tunnel vision
    b. Inattentional blindness
    c. Hyperacuity
    d. Binocular rivalry

    Explanation: Inattentional blindness or perceptual blindness occurs when an individual fails to perceive an unexpected stimulus in plain sight, purely as a result of a lack of attention rather than any vision defects or deficits

3. What is the main difference between simple cells and complex cells in the primary visual cortex (V1)?
    a. Simple cells primarily respond to color information, whereas complex cells primarily respond to luminance variations.
    b. Complex cells do not exhibit the same spatial segregation of ON and OFF regions as simple cells do.
    c. Simple cells respond to specific features such as edges and bars within their receptive fields. In contrast, complex cells respond to visual stimuli regardless of their exact location within the receptive field.
    d. Simple cells respond to simple visual features such as edges and bars. In contrast, complex cells are additionally able to detect movement while the simple cells cannot.

    Explanation: Simple cells primarily respond to specific features such as edges and bars within their receptive fields. They exhibit spatially organized responses with distinct ON and OFF regions. In contrast, complex cells integrate inputs from multiple simple cells and respond to more complex patterns of visual stimuli, such as oriented lines or bars, regardless of their exact location within the receptive field. Complex cells do not exhibit the same spatial segregation of ON and OFF regions as simple cells do.

4. Cortical magnification factor increases with eccentricity.
    a. True
    b. False

    Explanation: Cortical magnification factor describes how much cortical surface area is devoted for a unit of a visual field area. Cortical magnification factor decreases as the eccentricity increases as less resources are used for processing of the visual information from the periphery.

5. Extrastriate cortex includes:
    a. V1
    b. V2
    c. V3
    d. V4
    e. MT/V5
    f. IT
    g. LIP

    Explanation: The extrastriate cortex is the region of the occipital cortex of the mammalian brain located next to the primary visual cortex. Primary visual cortex (V1) is also named striate cortex because of its striped appearance in the microscope. The extrastriate cortex encompasses V3, V4, and MT/V5, while V1 corresponds to the striate cortex, and V2 to the prestriate cortex.

In [1]:
```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from scipy.signal import convolve2d
import math
```

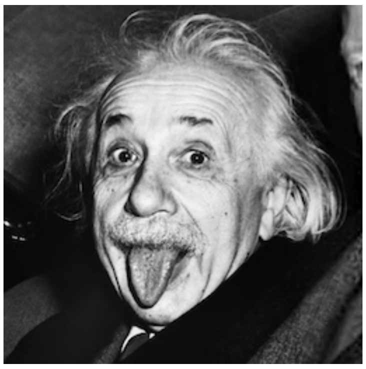
Homework 1

(A) Choosing an example image

In [2]:
```
def read_image(file):
    image = Image.open(file)

    # Convert to grayscale
    grey_avg_array = (np.sum(image, axis = -1, keepdims = False) / 3)
    grey_avg_array = grey_avg_array.astype(np.uint8)
    image = Image.fromarray(grey_avg_array, "L")

    return image
```

In [3]:
```
image = read_image("resources/einstein.jpeg")
image
```

Out[3]:



(B) Simple center-surround receptive field

Filter the photo by the center-surround receptive fields of the retinal ganglion cells. You can start with the practice of the toy models of the center-surround receptive fields that we used in the tutorial. These toy models have receptive field's shape described by $K(x, y)$ as a function of horizontal and vertical displacement x and y from the center of the receptive field.

$$K(x,y) = \begin{cases} 1 & \text{when } |x| < L/2, |y| < L/2 \\ -v & \text{when } |x| \geq L/2, |y| \geq L/2 \end{cases}$$

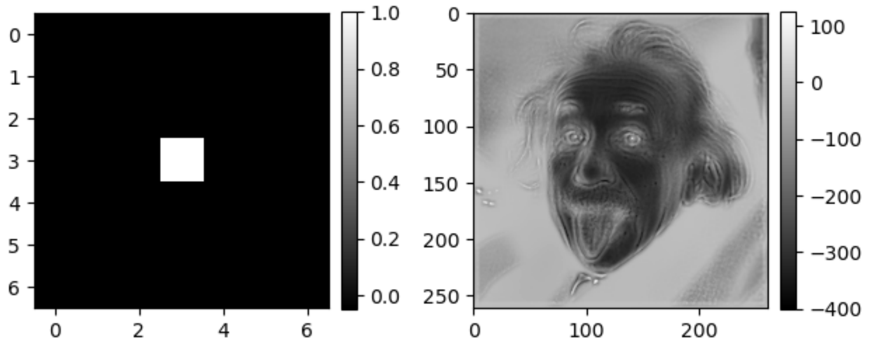
Give a couple of examples using different parameters L and v. Plot out the responses of the ganglion cells as an image to show the outcomes.

In [4]:
```
def square_filter(v, l):
    filter = np.ones(shape = (l, l)) * v

    # Estimate the center of the receptive field
    center = int(l / 2)

    # Estimate the width of the excitatory receptive field area
    width = int(l / 4)
    if width == 0: width = 1

    for i in range(filter.shape[0]):
        for j in range(filter.shape[1]):
            if np.abs(i - center) < width and np.abs(j - center) < width:
                filter[i, j] = 1

    return filter
```

In [5]:
```
def apply_filter(image, filter):
    return convolve2d(image, filter)
```

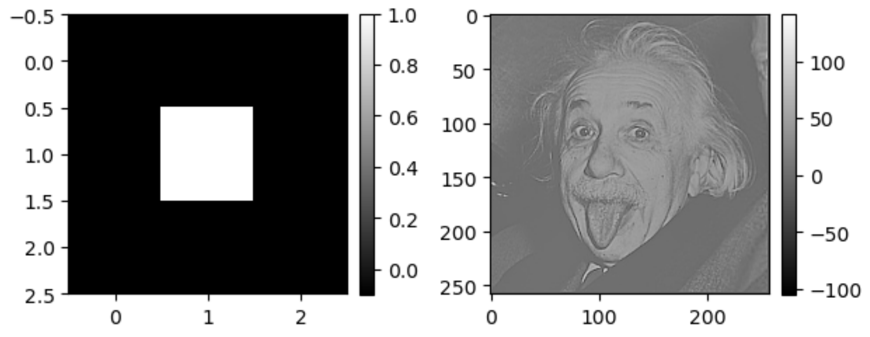
In [6]:
```
def plot_filter_and_image(filter, image):
    plt.subplot(1, 2, 1)
    plt.imshow(filter, cmap = "gray")
    plt.colorbar(fraction = 0.046, pad = 0.04)

    plt.subplot(1, 2, 2)
    plt.imshow(image, cmap = "gray")
    plt.colorbar(fraction = 0.046, pad = 0.04)

    plt.tight_layout()
    plt.show()
```

In [7]:
```
filter = square_filter(v = -0.05, l = 7)
plot_filter_and_image(filter, apply_filter(image, filter))
```



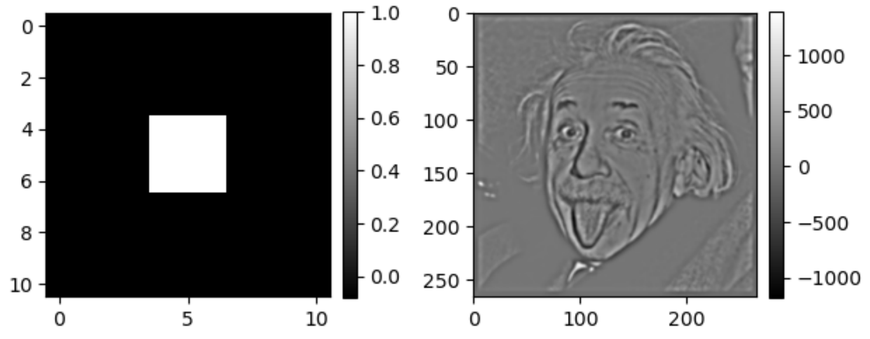
The image is filtered using $L = 7$ and $v = -0.05$. The colors are inverted because the average filter value is negative. Also, large receptive field makes the image more blurry.

In [8]:
```
filter = square_filter(v = -0.1, l = 3)
plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using $L = 3$ and $v = -0.1$. This time, the average filter value is positive and the colors are not inverted. Using a smaller receptive field maintains the acuity.

In [9]:
```
filter = square_filter(v = -9 / (11 * 11 - 9), l = 11)
plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using $L = 11$ and $v \approx -0.08$. This time, the v parameter calculated so that the average filter value is close to zero which results in non-inverted colors. This filter is able to detect edges within the image.

(C) Difference-of-Gaussians filter

Repeat (B) using a difference-of-gaussian $K(x, y)$ as

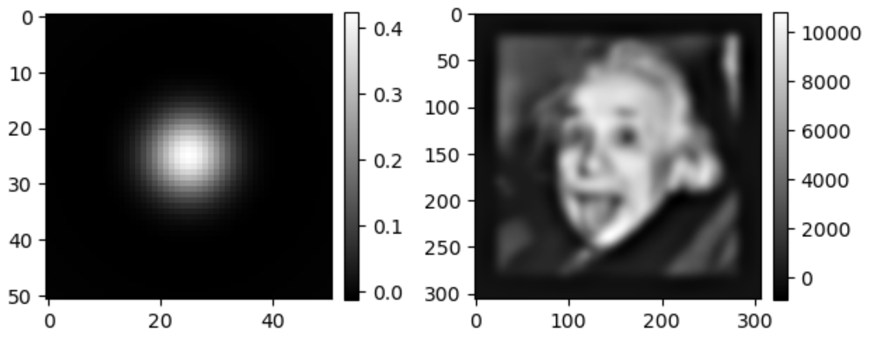
$$K(x,y) = \frac{w_c}{\sigma_c^2} exp(-\frac{x^2+y^2}{2\sigma_c^2}) - \frac{w_s}{\sigma_s^2} exp(-\frac{x^2+y^2}{2\sigma_s^2})$$

Play with parameters w_c, w_s, σ_c and σ_s and understand how $K(x, y)$ changes with each these parameters, and thus the meaning of these parameters. Filter the original image using this $K(x, y)$ and see the outcome as the retinal ganglion cells population responses. For a retinal ganglion cell, you may try $w_c = 1.1 w_s$ and $\sigma_s = 5\sigma_c$.

In [10]:
```
def gaussian(w_center, w_surround, sigma_center, sigma_surround, x, y):
    return (w_center / math.pow(sigma_center, 2)) * np.exp(-(math.pow(x, 2) + math.pow(y, 2)) / (2 * math.pow(s
        (w_surround / math.pow(sigma_surround, 2)) * np.exp(-(math.pow(x, 2) + math.pow(y, 2)) / (2 * math.pow(
```

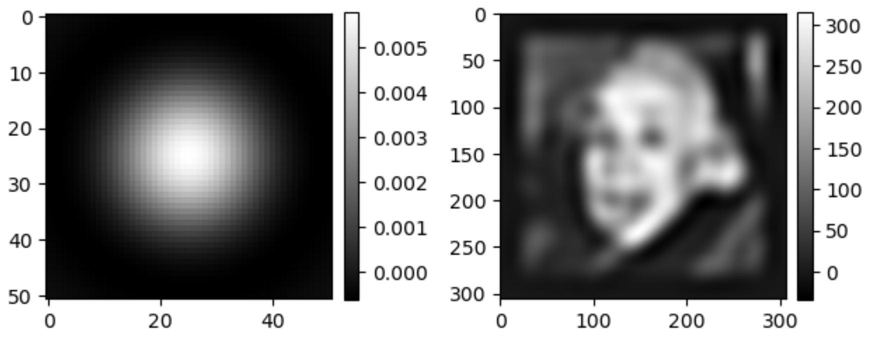
In [11]:
```
def gaussian_filter(w_center, w_surround, sigma_center, sigma_surround, l = 101):
    filter = np.zeros(shape = (l, l))

    center = int(l / 2)

    for i in range(filter.shape[0]):
        for j in range(filter.shape[1]):
            filter[ i, j ] = gaussian(w_center, w_surround, sigma_center, sigma_surround, i - center, j - cente

    return filter
```

In [12]:
```
filter = gaussian_filter(
    w_center = 11, # center strength
    sigma_center = 5, # center radius
    w_surround = 10, # surround strength
    sigma_surround = 25, # surround radius
    l = 51
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



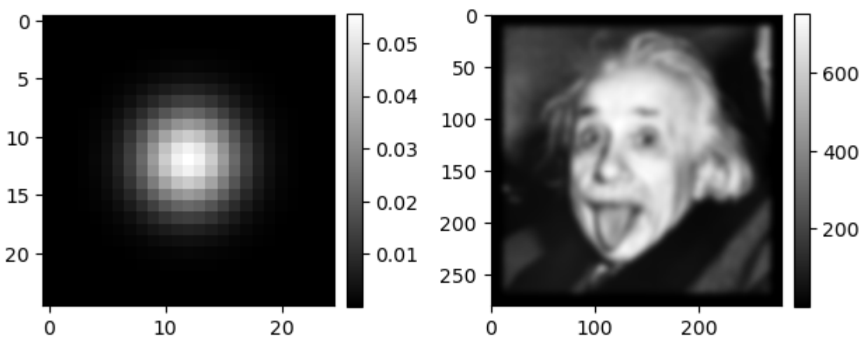
The image is filtered using $w_c = 11$, $w_s = 10$, $\sigma_c = 5$ and $\sigma_s = 25$. The w_c and w_s parameters control the **strength** of the center and the surround Gaussian, accordingly. The σ_c and σ_s parameters control the **size** of the center and the surround Gaussian, accordingly.

In [13]:
```
filter = gaussian_filter(
    w_center = 1, # center strength
    sigma_center = 10, # center radius
    w_surround = 0.95, # surround strength
    sigma_surround = 15, # surround radius
    l = 51
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



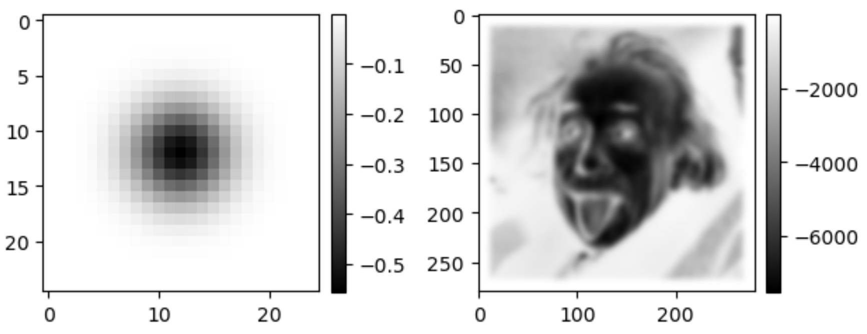
The image is filtered using $w_c = 1$, $w_s = 0.95$, $\sigma_c = 10$ and $\sigma_s = 15$.

In [14]:
```
filter = gaussian_filter(
    w_center = 0.5, # center strength
    sigma_center = 3, # center radius
    w_surround = 0, # surround strength
    sigma_surround = 1, # surround radius
    l = 25
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using $w_c = 0.5$, $w_s = 0$, $\sigma_c = 3$ and $\sigma_s = 1$ (only the center Gaussian is used).

In [15]:
```
filter = gaussian_filter(
    w_center = 0, # center strength
    sigma_center = 1, # center radius
    w_surround = 5, # surround strength
    sigma_surround = 3, # surround radius
    l = 25
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using $w_c = 0$, $w_s = 5$, $\sigma_c = 1$ and $\sigma_s = 3$ (only the surround Gaussian is used).

(D) Vertical orientation selective neuron

Repeat (C), but use a receptive field filter shape $K(x, y)$ that models a V1's simple cell's receptive field. Use an orientation selective neuron (tuning to vertical orientation), with

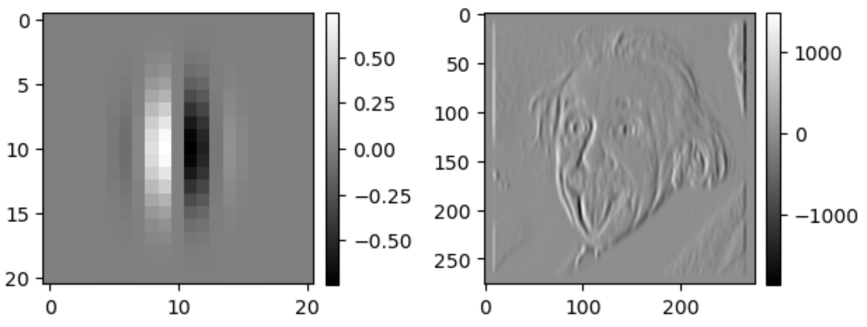
$$K(x, y) = exp(-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2})cos(\bar{k}x + \phi)$$

Vary parameters σ_x, σ_y, $\bar{k}$, and ϕ and plot out $K(x, y)$ using different set of parameters to see how these parameters control the shape of $K(x, y)$. For a V1 cell model, try for example $\sigma_y = 1.5\sigma_x$, and $\bar{k} = 2\pi/(3\sigma_x)$, and filter the original image using this $K(x, y)$ and see the outcome as the population responses of V1 neurons preferring this orientation.

In [16]:
```
def vertical_orientation(sigma_x, sigma_y, k, phi, x, y):
    return np.exp(-(math.pow(x, 2) / (2 * math.pow(sigma_x, 2))) - (math.pow(y, 2)) / (2 * math.pow(sigma_y, 2)
```

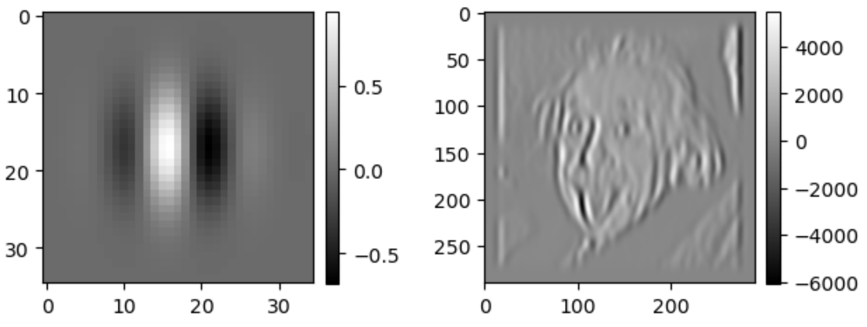
In [17]:
```
def vertical_filter(sigma_x, sigma_y, k, phi, l = 101):
    filter = np.zeros(shape = (l, l))

    center = int(l / 2)

    for i in range(filter.shape[0]):
        for j in range(filter.shape[1]):
            filter[i, j] = vertical_orientation(sigma_x, sigma_y, k, phi, j - center, i - center)
    return filter
```

In [18]:
```
filter = vertical_filter(
    sigma_x = 2,
    sigma_y = 3,
    k = 1,
    phi = math.radians(90),
    l = 21
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



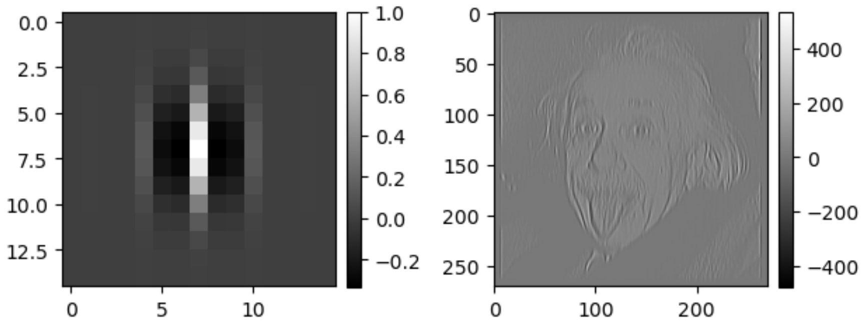
The image is filtered using $\sigma_x = 2$, $\sigma_y = 3$, $\bar{k} = 1$, and $\phi = 90^\circ$.

In [19]:
```
filter = vertical_filter(
    sigma_x = 5,
    sigma_y = 5,
    k = 0.5,
    phi = math.radians(45),
    l = 35
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



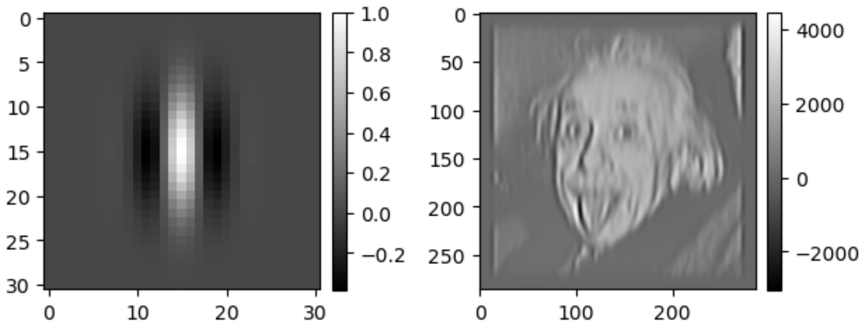
The image is filtered using $\sigma_x = 5$, $\sigma_y = 5$, $\bar{k} = 0.5$, and $\phi = 45^\circ$.

In [20]:
```
filter = vertical_filter(
    sigma_x = 1.5,
    sigma_y = 2,
    k = 2,
    phi = math.radians(0),
    l = 15
)

plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using $\sigma_x = 1.5$, $\sigma_y = 2$, $\bar{k} = 2$, and $\phi = 0^\circ$.

```
In [21]: filter = vertical_filter(
             sigma_x = 3,
             sigma_y = 3 * 1.5,
             k = (2 * math.pi) / (3 * 3),
             phi = math.radians(0),
             l = 31
         )

         plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using $\sigma_x = 3$, $\sigma_y = 1.5\sigma_x$, $\bar{k} = 2\pi/(3\sigma_x)$, and $\phi = 0°$.

(E) Horizontal orientation selective neuron

Repeat (D), but using a $K(x, y)$ that is tune to horizontal orientation, i.e.,

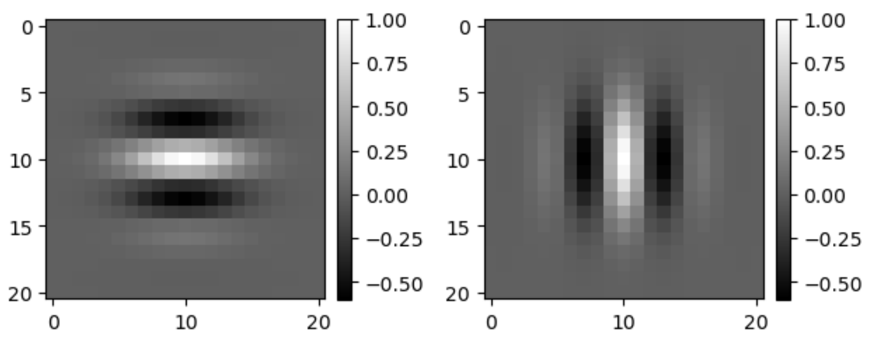
$$K(x, y) = exp(-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2})cos(\bar{k}y + \phi)$$

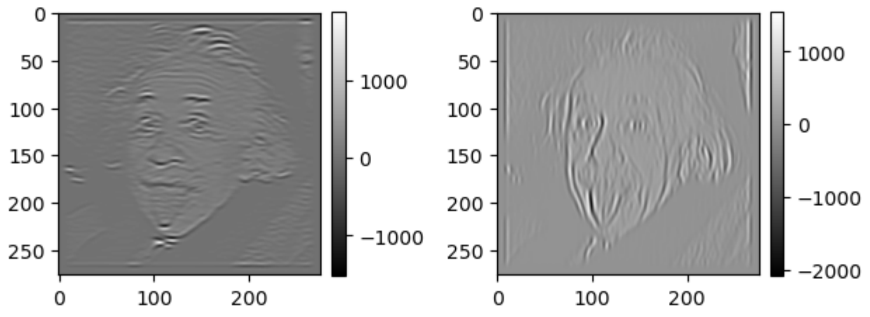
Plot out $K(x, y)$ from equations 3 and 4 side by side to compare and see they how differ from each other. Plot out the neural population responses from these two filters side by side and see how they differ from each other, and comment on these differences.

```
In [22]: def horizontal_orientation(sigma_x, sigma_y, k, phi, x, y):
             return np.exp(-(math.pow(x, 2) / (2 * math.pow(sigma_x, 2))) - (math.pow(y, 2)) / (2 * math.pow(sigma_y, 2)
```

```
In [23]: def horizontal_filter(sigma_x, sigma_y, k, phi, l = 101):
             filter = np.zeros(shape = (l, l))

             center = int(l / 2)

             for i in range(filter.shape[0]):
                 for j in range(filter.shape[1]):
                     filter[i, j] = horizontal_orientation(sigma_x, sigma_y, k, phi, j - center, i - center)

             return filter
```

```
In [24]: filter_horizontal = horizontal_filter(
             sigma_x = 3,
             sigma_y = 3,
             k = 1,
             phi = math.radians(0),
             l = 21
         )

         filter_vertical = vertical_filter(
             sigma_x = 3,
             sigma_y = 3,
             k = 1,
             phi = math.radians(0),
             l = 21
         )

         plot_filter_and_image(filter_horizontal, filter_vertical)
         plot_filter_and_image(apply_filter(image, filter_horizontal), apply_filter(image, filter_vertical))
```



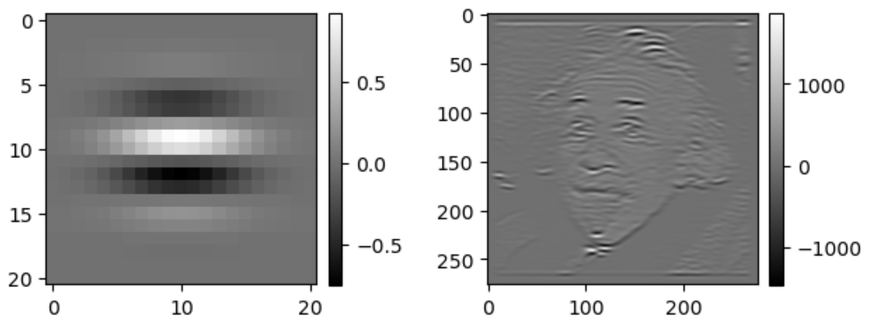


The left image is filtered using a horizontal filter with parameters $\sigma_x = 3$, $\sigma_y = 3$, $\bar{k} = 1$, and $\phi = 0°$.

The right image is filtered using a vertical filter with parameters $\sigma_x = 3$, $\sigma_y = 3$, $\bar{k} = 1$, and $\phi = 0°$.

The two filters are rotated 90 degrees in respect to each other. The vertically tuned filter detects vertical lines, and the horizontally tuned filter detects horizontal lines. Both filters blur other features in the picture.

```
In [25]: filter = horizontal_filter(
             sigma_x = 3.5,
             sigma_y = 3,
             k = 1,
             phi = math.radians(45),
             l = 21
         )

         plot_filter_and_image(filter, apply_filter(image, filter))
```



The image is filtered using a filter with parameters $\sigma_x = 3.5$, $\sigma_y = 3$, $\bar{k} = 1$, and $\phi = 45°$.

(F) Contrast sensitivity function for a retinal ganglion cell

Use the retinal ganglion cell's $K(x, y)$ from part (C) and try to get its contrast sensitivity function $g(k)$ by calculating

$$g_c(k) = \sum_{x,y} K(x, y)cos(kx)$$

and

$$g_s(k) = \sum_{x,y} K(x, y)sin(kx)$$

Finally, calculate

$$g(k) = \sqrt{[g_s(k)]^2 + [g_c(k)]^2}$$

for all kinds of values of k, so that you get $g(k)$ as a function of k, and plot out $g(k)$ versus k. Do you see that $g(k)$ peaks at a particular k?

```
In [26]: def g_c(kernel, k):
             res = 0

             for i in range(kernel.shape[0]):
                 for j in range(kernel.shape[1]):
                     pos = j
                     res += kernel[i, j] * np.cos(k * pos)

             return res
```

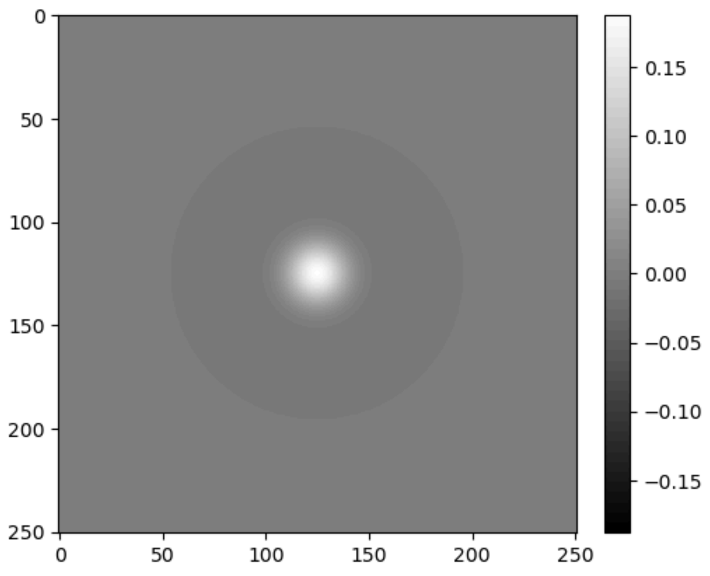
```
In [27]: def g_s(kernel, k):
             res = 0

             for i in range(kernel.shape[0]):
                 for j in range(kernel.shape[1]):
                     pos = j
                     res += kernel[i, j] * np.cos(k * pos)

             return res
```

```
In [28]: def g(kernel, k):
             g_k = []
             for _k in k:
                 gs = math.pow(g_s(kernel, _k), 2)
                 gc = math.pow(g_c(kernel, _k), 2)
                 g_k.append(math.sqrt(gs + gc))

             return g_k
```

```
In [29]: def k_step(pixels):
             return [ n * 2 * math.pi / pixels for n in range(0, int(pixels / 2)) ]
```

```
In [30]: kernel = gaussian_filter(
             w_center = 20, # center strength
             sigma_center = 10, # center radius
             w_surround = 30, # surround strength
             sigma_surround = 10 * 5, # surround radius
             l = 251
         )
```

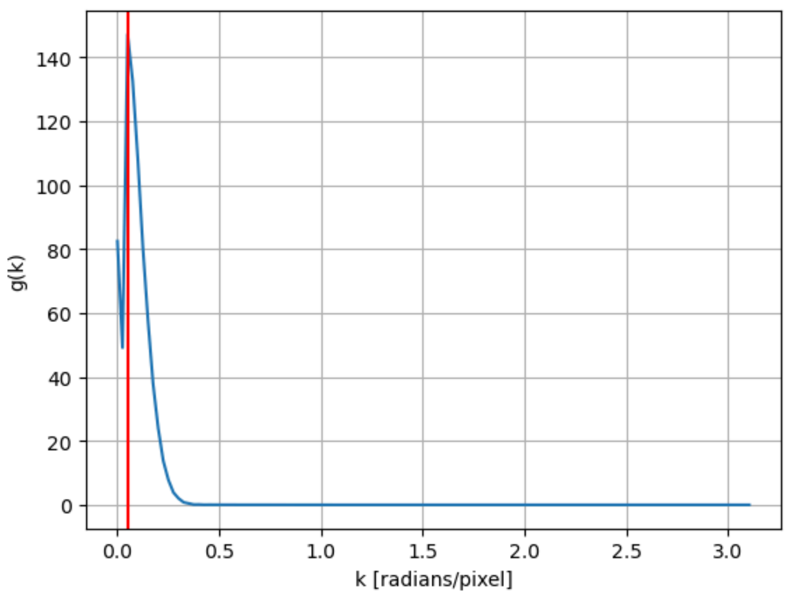
```
In [31]: plt.imshow(kernel, vmin = -np.max(np.abs(kernel)), vmax = np.max(np.abs(kernel)), cmap = "gray")
         plt.colorbar(fraction = 0.046, pad = 0.04)
         plt.show()
```



Above is a difference-of-Gaussians filter with parameters $w_c = 20$, $w_s = 30$, $\sigma_c = 10$ and $\sigma_s = 50$.

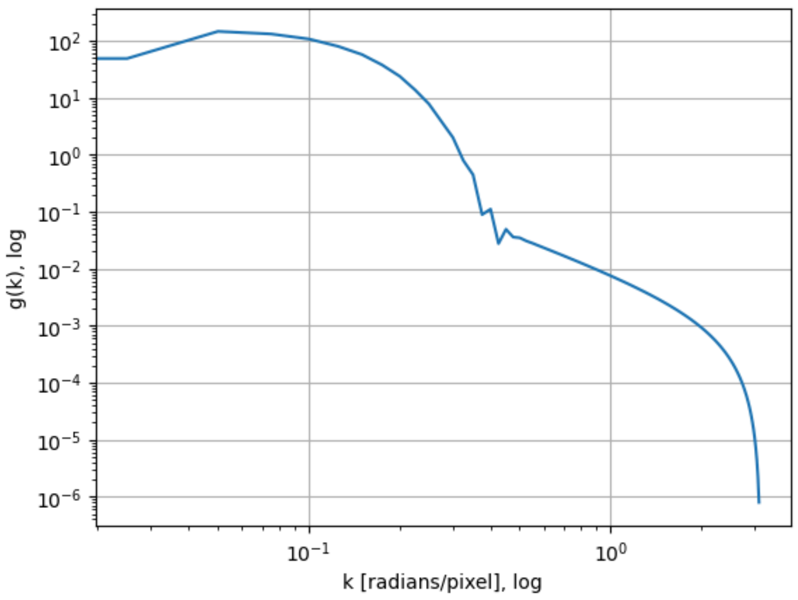
```
In [32]: k = k_step(251)
         g_k = g(kernel, k)
```

```
In [33]: plt.plot(k, g_k)
         plt.axvline(0.05, color = "red")
         plt.grid()
         plt.xlabel("k [radians/pixel]")
         plt.ylabel("g(k)")
         plt.show()
```



Contrast sensitivity function of a retinal ganglion cell (difference-of-Gaussians filter) peaks at $k \approx 0.05$.

In [34]:
```python
plt.plot(k, g_k)
plt.grid()
plt.xlabel("k [radians/pixel], log")
plt.ylabel("g(k), log")
plt.xscale("log")
plt.yscale("log")
plt.show()
```

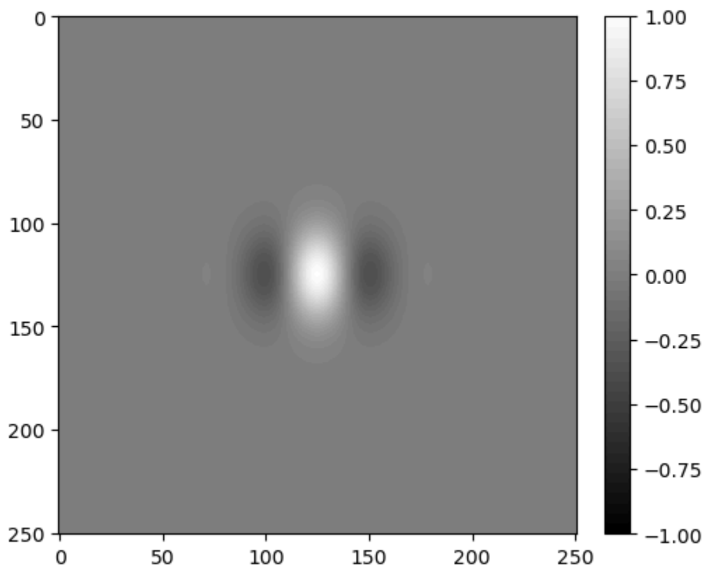


(G) Contrast sensitivity function for V1 cell

Repeat part (F) with V1's receptive field, i.e., use $K(x, y)$ in part (D), and get $g(k)$. Which k value is this $g(k)$ peaking at? Is it around $k = \bar{k}$? Compare this $g(k)$ with the one you have in part (F), and see how they differ from each other.

In [35]:
```python
kernel = vertical_filter(
    sigma_x = 20,
    sigma_y = 15,
    k = 0.1,
    phi = math.radians(0),
    l = 251
)
```

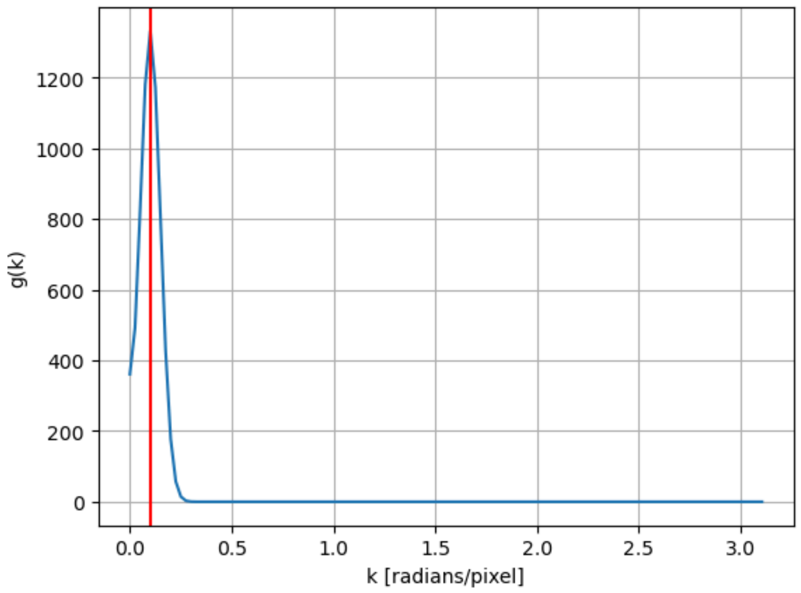
In [36]:
```python
plt.imshow(kernel, cmap = "gray", vmin = -np.max(np.abs(kernel)), vmax = np.max(np.abs(kernel)))
plt.colorbar(fraction = 0.046, pad = 0.04)
plt.show()
```



Above is a vertical filter with parameters $\sigma_x = 20$, $\sigma_y = 15$, $\bar{k} = 0.1$, and $\phi = 0^\circ$.

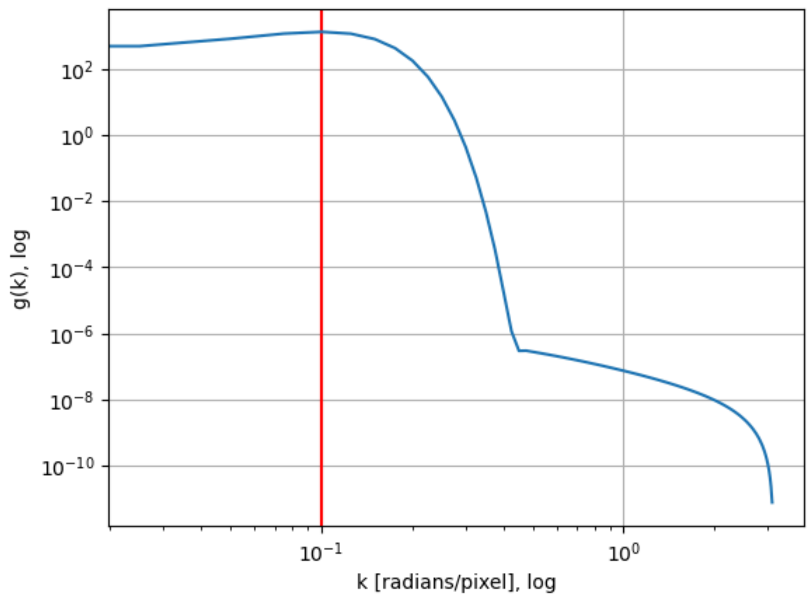
```
In [37]: k_vertical = k_step(251)
         g_k_vertical = g(kernel, k_vertical)
```

```
In [38]: plt.plot(k_vertical, g_k_vertical)
         plt.grid()
         plt.axvline(0.1, color = "red")
         plt.xlabel("k [radians/pixel]")
         plt.ylabel("g(k)")
         plt.show()
```

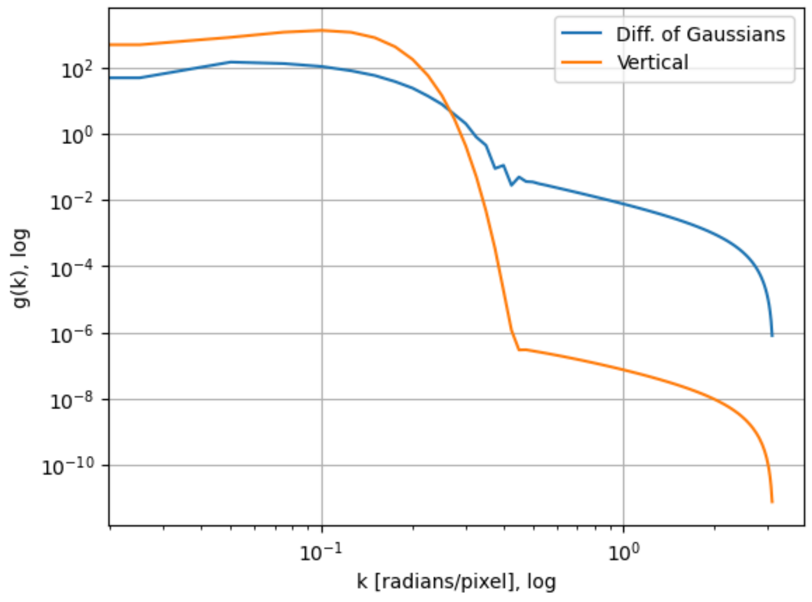


Contrast sensitivity function of a vertical filter peaks at $k \approx 0.1 = \bar{k}$.

```
In [39]: plt.axvline(0.1, color = "red")
         plt.plot(k_vertical, g_k_vertical)
         plt.grid()
         plt.xlabel("k [radians/pixel], log")
         plt.ylabel("g(k), log")
         plt.xscale("log")
         plt.yscale("log")
         plt.show()
```



```
In [40]: plt.plot(k, g_k, label = "Diff. of Gaussians")
         plt.plot(k_vertical, g_k_vertical, label = "Vertical")
         plt.grid()
         plt.xlabel("k [radians/pixel], log")
         plt.ylabel("g(k), log")
         plt.xscale("log")
         plt.yscale("log")
         plt.legend()
         plt.show()
```



The contrast sensitivity function of the difference-of-Gaussians filter has a less pronounced drop as k increases. This is due to there being a difference of two Gaussian functions that peak at different values of k. The contrast sensitivity function of the vertical filter, on the other hand, has a stronger peak and then drops fast.

(H) Power spectrum of an image

Repeat part (G) by replacing $K(x, y)$ by your own original image. Let us denote your original image as $S(x, y)$ as a function of x and y.

$$S_c(k) = \sum_{x,y} S(x, y)cos(kx)$$

and

$$S_s(k) = \sum_{x,y} S(x, y)sin(kx)$$

From $S_c(k)$ and $S_s(k)$ calculate

$$|S(k)|^2 = [S_c(k)]^2 + [S_s(k)]^2$$

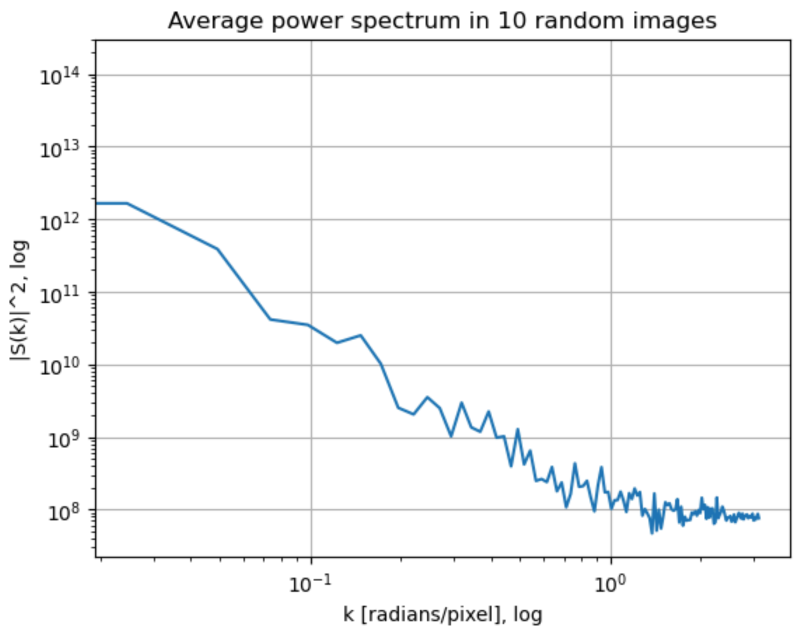
This $|S(k)|^2$ as a function of k is called the power spectrum of an image. Plot out this function, and note a general trend of how $|S(k)|^2$ changes with k.

```
In [41]: def power_spectrum(image, k):
             g_k = []
             for _k in k:
                 gs = math.pow(g_s(image, _k), 2)
                 gc = math.pow(g_c(image, _k), 2)
                 g_k.append(gs + gc)

             return g_k
```

```
In [42]: image_files = [ f"resources/image{i}.jpeg" for i in range(1, 10) ]
         image_files = np.append(image_files, "resources/einstein.jpeg")

         k = k_step(256)
         s_k = []

         for file in image_files:
             print(f"Processing file {file}")
             image = read_image(file)
             s_k.append(power_spectrum(np.asarray(image, dtype = "uint8"), k))

         s_k = np.array(s_k)
```

```
Processing file resources/image1.jpeg
Processing file resources/image2.jpeg
Processing file resources/image3.jpeg
Processing file resources/image4.jpeg
Processing file resources/image5.jpeg
Processing file resources/image6.jpeg
Processing file resources/image7.jpeg
Processing file resources/image8.jpeg
Processing file resources/image9.jpeg
Processing file resources/einstein.jpeg
```

```
In [43]: plt.plot(k, np.mean(s_k, axis = 0))
         plt.grid()
         plt.xlabel("k [radians/pixel], log")
         plt.ylabel("|S(k)|^2, log")
         plt.xscale("log")
         plt.yscale("log")
         plt.title("Average power spectrum in 10 random images")
         plt.show()
```



In general, the power drops as the k increases.

```
In [43]:
```