In [2]:
```
# ! unzip imgs.zip
```

In [5]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import os
```

Read image dataset

In [6]:
```
folder_path = "imgs/"
```

In [7]:
```
image_files = [f for f in os.listdir(folder_path) if f.endswith('.png')]
```

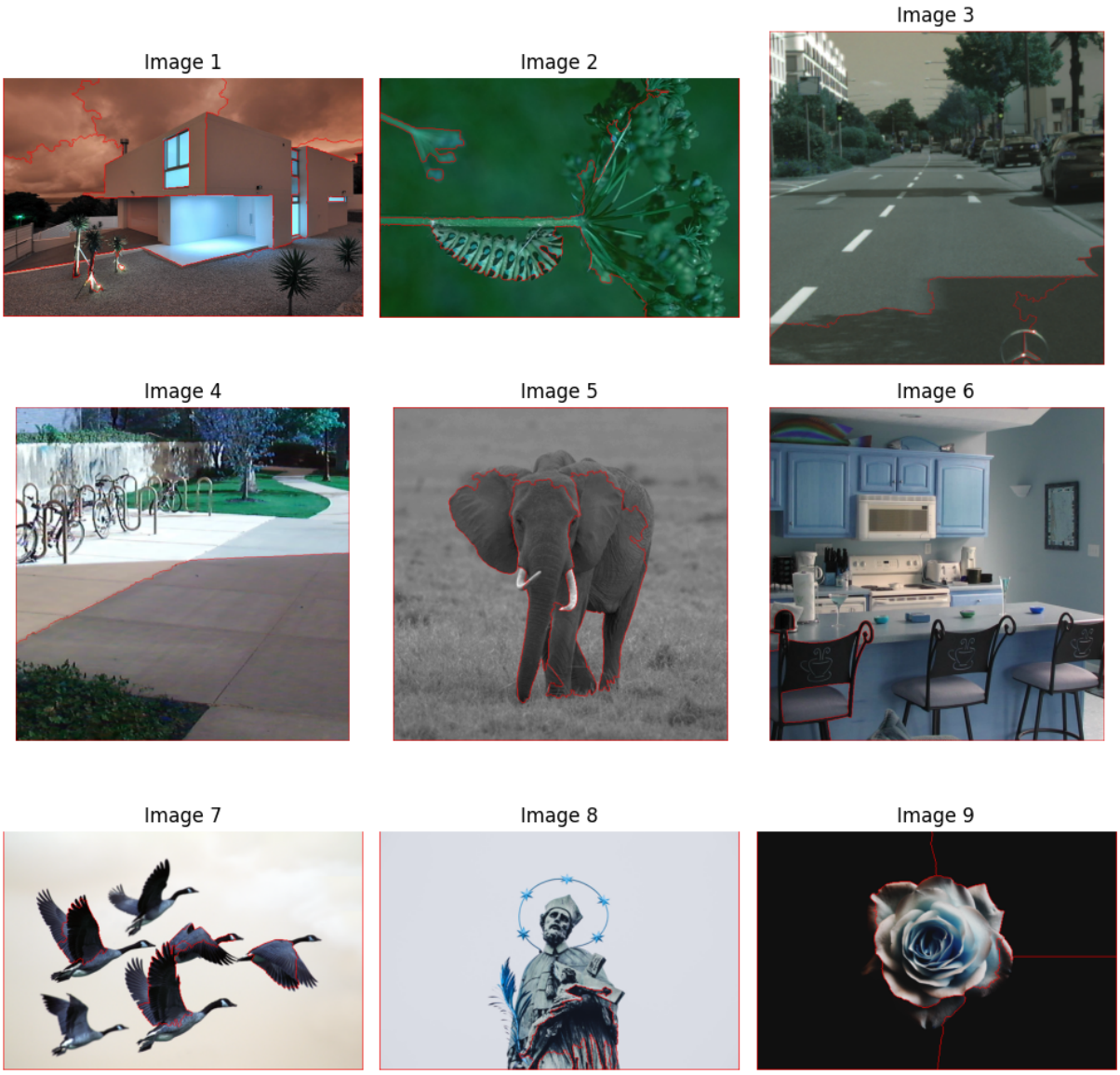
Watershed segmentation

In [12]:
```
fig, axes = plt.subplots(3, 3, figsize=(10, 10))
axes = axes.flatten()
for i, ax in enumerate(axes):
      image = cv2.imread(os.path.join(folder_path, image_files[i]))
      gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
      ret, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY_INV + cv2.
      kernel = np.ones((3, 3), np.uint8)
      opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=
      sure_bg = cv2.dilate(opening, kernel, iterations=3)
      dist_transform = cv2.distanceTransform(opening, cv2.DIST_L2, 5)
      ret, sure_fg = cv2.threshold(dist_transform, 0.7 * dist_transform.max(
      sure_fg = np.uint8(sure_fg)
      unknown = cv2.subtract(sure_bg, sure_fg)
      ret, markers = cv2.connectedComponents(sure_fg)
      markers = markers + 1
      markers[unknown == 255] = 0
      markers = cv2.watershed(image, markers)
      image[markers == -1] = [255, 0, 0]

      ax.imshow(image)
      ax.axis('off')
      ax.set_title(f'Image {i+1}')

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

Loading [MathJax]/extensions/Safe.js



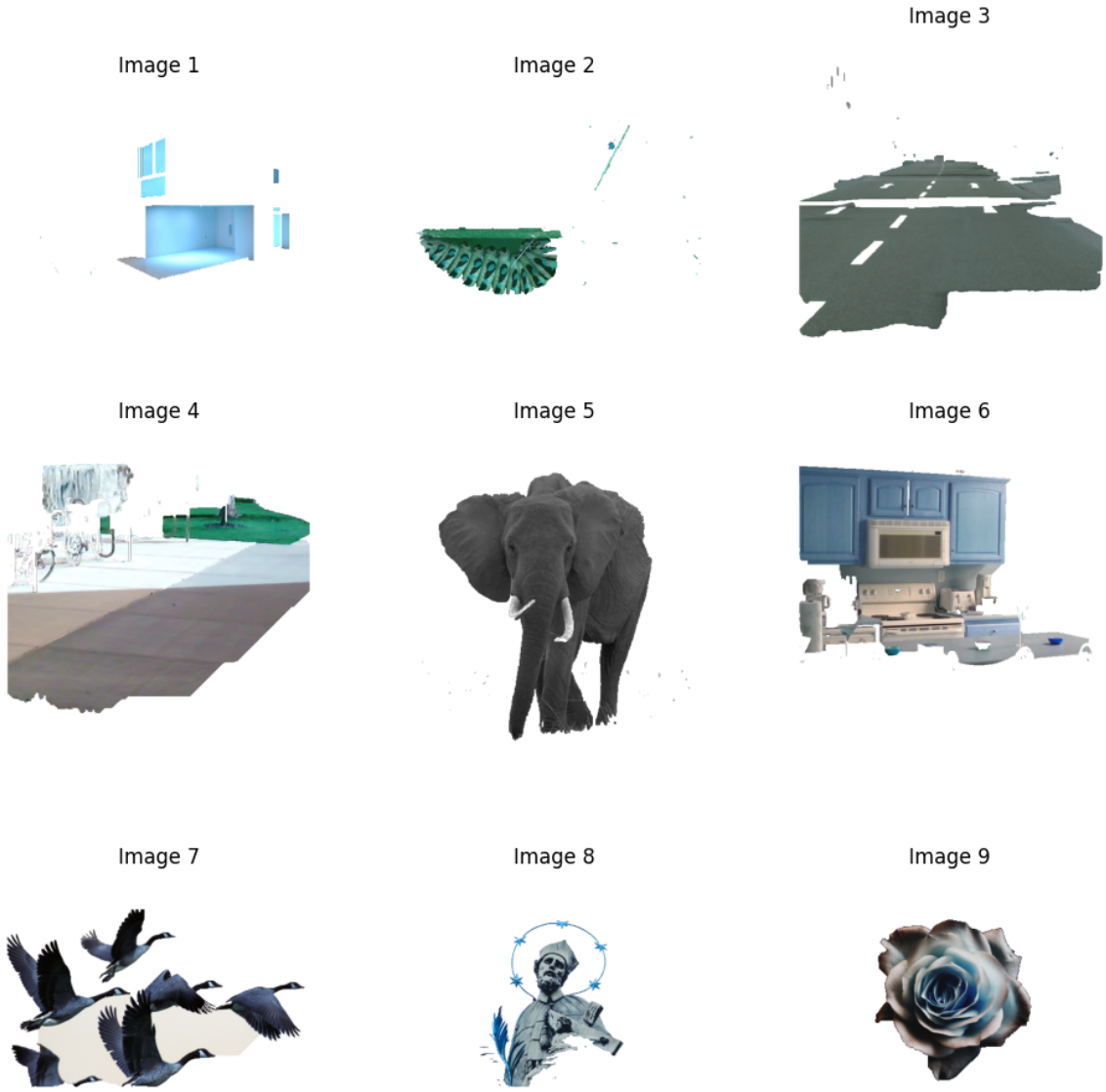
GrabCut segmentation

In [13]:
```
fig, axes = plt.subplots(3, 3, figsize=(10, 10))
axes = axes.flatten()
for i, ax in enumerate(axes):
      image = cv2.imread(os.path.join(folder_path, image_files[i]))
      mask = np.zeros(image.shape[:2], np.uint8)
      bgd_model = np.zeros((1, 65), np.float64)
      fgd_model = np.zeros((1, 65), np.float64)
      rect = (50, 50, image.shape[1] - 100, image.shape[0] - 100)
      cv2.grabCut(image, mask, rect, bgd_model, fgd_model, 5, cv2.GC_INIT_WI
      mask2 = np.where((mask == 2) | (mask == 0), 0, 1).astype('uint8')
      segmented_image = image * mask2[:, :, np.newaxis]
      segmented_image[np.where((segmented_image == [0, 0, 0]).all(axis=2))]

      ax.imshow(segmented_image)
      ax.axis('off')
      ax.set_title(f'Image {i+1}')
```

Loading [MathJax]/extensions/Safe.js

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



Loading [MathJax]/extensions/Safe.js