Araba fotoğrafını medyan filtreden geçirerek gürültüden temizliyoruz.

```
original_img = cv2.imread('I.png')
original_img_rgb = cv2.cvtColor(original_img, cv2.COLOR_BGR2RGB)

median_img = median_filter_salt_and_pepper(original_img,5)
filtered_image_rgb = cv2.cvtColor(median_img, cv2.COLOR_BGR2RGB)

print("Image shape:", original_img.shape, "dtype:", original_img.dtype)

fig, axes = plt.subplots(1, 2, figsize=(10, 5))
axes[0].imshow(original_img_rgb)
axes[0].set_title('Orijinal Fotoğraf')
axes[0].axis('off')

axes[1].imshow(filtered_image_rgb)
axes[1].set_title('Medyan Filtreli Fotoğraf')
axes[1].axis('off')
```

Kenar görüntülemeyi kolaylaştırmak için önce gri tonlama ve gauss filtresi uyguluyoruz. Kenar çıktısı için Canny kenar tespiti uyguluyoruz.

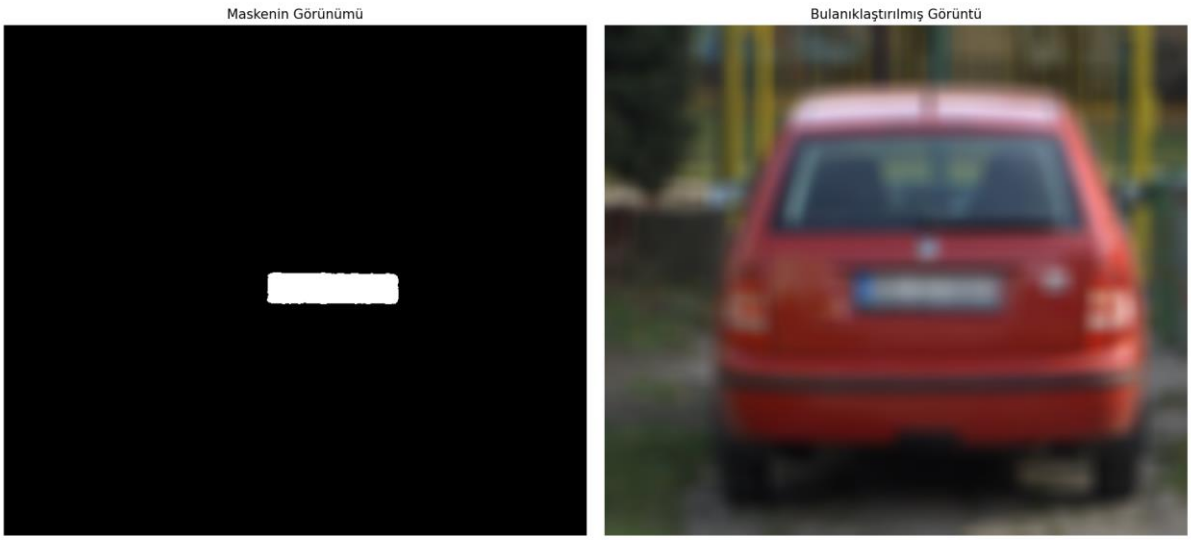
```
gray = cv2.cvtColor(original_img, cv2.COLOR_BGR2GRAY)

gauss = cv2.GaussianBlur(gray, (5, 5), 3)

edges = cv2.Canny(gauss, 40, 80)

fig, axes = plt.subplots(1, 3, figsize=(10, 5))
axes[0].imshow(gray, cmap='gray')
axes[0].set_title('Gri Tonlama ve Gürültü Azaltma')
axes[0].axis('off')

axes[1].imshow(gauss, cmap='gray')
axes[1].set_title('Gauss')
axes[1].axis('off')

axes[2].imshow(edges, cmap='gray')
axes[2].set_title('Canny Kenar Tespiti')
axes[2].axis('off')

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

Canny kenar bulmadan sonra konturları buluyoruz

```
1. contours, _ = cv2.findContours(edges, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
2. print("Tespit edilen kontur sayısı:", len(contours))
3.
4. plt.imshow(cv2.drawContours(filtered_image_rgb, contours, -1, (0, 255, 0), 3), cmap="gray")
5. plt.title('Konturların Çizilmesi')
6. plt.axis('off')
7. plt.show()
```

Plaka olabilecek konturların en boy oranı ve kapladıkları alana göre ‘plaka adayları’ dizinine ekliyoruz. Koşulları sağlayan konturla beyaz maske oluşturup bu bölgeye gauss ile bulanıklaştırma uyguluyoruz.

```
plate_candidates = []
for contour in contours:
    x, y, w, h = cv2.boundingRect(contour)
    aspect_ratio = w / float(h)
    area = cv2.contourArea(contour)
    print(f"Alan: {area}, En-Boy Oranı: {aspect_ratio}")

    if (area > 500 and area < 100000) and (aspect_ratio > 1.5 and aspect_ratio < 6):
        plate_candidates.append(contour)

        mask = np.zeros_like(original_img)
        cv2.drawContours(mask, [contour], -1, (255, 255, 255), thickness=cv2.FILLED)

        blurred = cv2.GaussianBlur(original_img, (25, 25), 30)

        original_img = np.where(mask == 255, blurred, original_img)

mask_img = np.zeros_like(original_img)
cv2.drawContours(mask_img, plate_candidates, -1, (255, 255, 255), -1)

temp_rgb = cv2.cvtColor(original_img, cv2.COLOR_BGR2RGB)
blurred = cv2.GaussianBlur(temp_rgb, (25, 25), 30)

fig, axes = plt.subplots(1, 2, figsize=(10, 5))
axes[0].imshow(mask_img)
axes[0].set_title('Maskenin Görünümü')
axes[0].axis('off')

axes[1].imshow(blurred)
axes[1].set_title('Bulanıklaştırılmış Görüntü')
axes[1].axis('off')

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



Sonuç

```
1. original_img_rgb = cv2.cvtColor(original_img, cv2.COLOR_RGB2BGR)
2. plt.imshow(original_img_rgb)
3. plt.title('Sonuç')
4. plt.axis('off')
5. plt.show()
```