```
        labels = []

        for _, row in batch_data.iterrows():
            image_path = f"/kaggle/input/augmented-forest
            image = cv2.imread(image_path)

            if image is None:
                print(f"Warning: Missing image at {image_
                continue

            image = cv2.resize(image, self.img_size)
            image = image / 255.0  # Normalize
            images.append(image)
            labels.append(1 if row['label'] == 'forested'

        if not images:
            return np.zeros((self.batch_size, *self.img_s

        images = np.array(images, dtype=np.float32)
        labels = np.array(labels, dtype=np.int32)

        if self.augment:
            images = next(self.datagen.flow(images, batch

        return images, labels

    def on_epoch_end(self):
        if self.shuffle:
            self.dataframe = self.dataframe.sample(frac=1
```

+ Code + Markdown

[54]:

```
train_generator = ImageGenerator(train_data, augment=True
test_generator = ImageGenerator(test_data, augment=False)

# Sanity check: Visualize a batch
sample_images, sample_labels = train_generator[0]
print(f"Batch shape: {sample_images.shape}")
print(f"Labels: {sample_labels[:5]}")

plt.figure(figsize=(10, 5))
for i in range(3):
    plt.subplot(1, 3, i+1)
    plt.imshow(sample_images[i])
```