In [1]:

```python
import cv2
import numpy as np
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
from skimage.feature import hog
import os

def load_images_from_folder(folder):
    images = []
    for filename in os.listdir(folder):
        img = cv2.imread(os.path.join(folder, filename), cv2.IMREAD_GRAYSCALE)
        if img is not None:
            images.append(img)
    return images

def resize_images(images, new_size=(64, 64)):
    resized_images = [cv2.resize(img, new_size) for img in images]
    return resized_images

# Update the paths for the training dataset
train_defective_path = r'C:\Users\Dell\Desktop\Python\Pepsico RnD Potato Lab Datase
train_non_defective_path = r'C:\Users\Dell\Desktop\Python\Pepsico RnD Potato Lab Da

# Load training images and labels
train_defective_images = load_images_from_folder(train_defective_path)
train_non_defective_images = load_images_from_folder(train_non_defective_path)

# Resize training images
train_defective_images_resized = resize_images(train_defective_images)
train_non_defective_images_resized = resize_images(train_non_defective_images)

# Assign labels (1 for defective, -1 for non-defective)
train_defective_labels = -1 * np.ones(len(train_defective_images_resized))
train_non_defective_labels = np.ones(len(train_non_defective_images_resized))

# Combine training images and labels
X_train = train_defective_images_resized + train_non_defective_images_resized
y_train = np.concatenate([train_defective_labels, train_non_defective_labels])

# Extract HOG features for each training image
hog_features_train = [hog(img, orientations=8, pixels_per_cell=(16, 16), cells_per_
X_train_hog = np.array(hog_features_train)

# Create and train the Support Vector Regressor (SVR)
svr = SVR(kernel='linear', C=1.0)  # You can choose different kernels and adjust pa
svr.fit(X_train_hog, y_train)

# Update the single path for the testing dataset
test_dataset_path = r'C:\Users\Dell\Desktop\Python\Pepsico RnD Potato Lab Dataset\T

# Load testing images from the combined dataset
test_images = load_images_from_folder(test_dataset_path)

# Resize testing images
test_images_resized = resize_images(test_images)

# Extract HOG features for each testing image
hog_features_test = [hog(img, orientations=8, pixels_per_cell=(16, 16), cells_per_b
X_test_hog = np.array(hog_features_test)

# Make predictions on the test set
y_pred = svr.predict(X_test_hog)
```

```

# Create dummy labels for y_test (you should replace this with your actual labels)
y_test = np.ones(len(y_pred))

# Display the number of defective and non-defective pieces in the combined test set
num_defective = np.sum(y_pred < 0)
num_non_defective = np.sum(y_pred >= 0)
print(f"Number of Defective Pieces: {num_defective}")
print(f"Number of Non-Defective Pieces: {num_non_defective}")

# Evaluate the model using mean squared error
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse:.4f}")
```

```
Number of Defective Pieces: 4
Number of Non-Defective Pieces: 4
Mean Squared Error: 1.5556
```

In [ ]: