

資 料

瓦の品質管理を目的とした瓦の外観検査システムの開発

藤原 直樹^{* **}・平井 克尚^{* ***}

1. 目 的

島根県の西部地域で生産されている石州瓦の表面には釉薬層が形成されており、銀黒色、来待色、赤色の瓦が主に販売されている。

その石州瓦の製造における課題の1つに、全ての瓦に対して目視により釉薬層の傷、亀裂、素地の露出の有無を確認する検査工程の作業負担低減がある。瓦は連続的に検査工程にコンベアで運ばれてくるため、検査担当者は検査工程に長時間常駐して1枚の瓦に対して数秒で目視検査を行う必要があり、作業者の負担は大きい。そこで瓦会社と共同で検査工程の自動化に取り組むこととした。検査工程の自動化を実現する手段の1つとして、機械学習（以下 ML と略す）を手法とした画像処理が挙げられる。特にディープラーニングを利用したリアルタイムの画像処理をシステム化することで、数秒での目視検査が要求される瓦の検査工程の自動化が期待できる。

本報では、ML を用いた瓦の良否を判定する画像処理システムを構築し、構築した画像処理システムによる瓦の良否判定性能精度と判定に要する時間を検証した結果について報告する。

2. 方 法

2.1 瓦の外観検査システムの仕様および要件

2.1.1 機能仕様

瓦の外観検査システムの機能仕様を表1に示す。表1のとおり、機能仕様は瓦画像の撮影機能と瓦の良否判定機能の2つに大別できる。本資料で述べる画像処理システムは、表1に示した機能項目の1つである瓦の良否判定機能をシングルボードコンピュータ（以下 SBC と略す）へ実装したものである。SBC は1つの基板上に CPU、メインメモリ、入出力端子等が実装された小型のコンピュータである。

2.1.2 不良判定基準

構築する画像処理システムでは、傷、亀裂、素地の露出を不良判定基準として、瓦を良品と不良品の2つに分類する。ここで、良品の瓦と、傷、亀裂、素地の露出により不

表1 瓦の外観検査システムの機能仕様

機能項目	機能概要
瓦画像の撮影機能	製造ラインを通過する瓦を撮影する機能
瓦の良否判定機能	撮影された瓦画像の良否を判定する機能

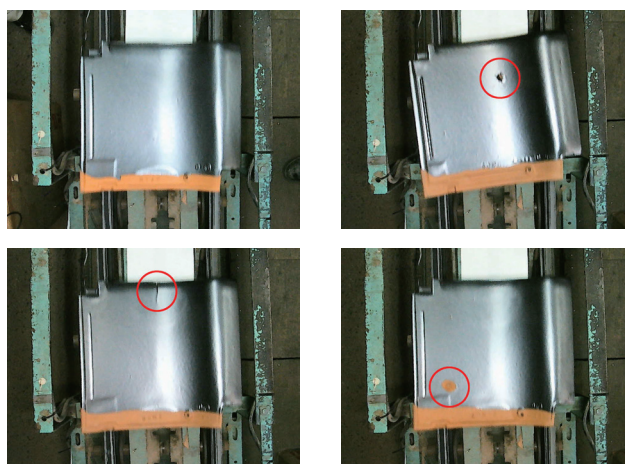


図1 代表的な良品画像および不良品画像
(左上：良品，右上：傷，左下：亀裂，右下：素地の露出。
画像内赤枠の箇所が該当する不良箇所。)

良と判定した瓦を図1に示す。

2.1.3 システム要件

筆者らが課題聴取を実施した石州瓦の製造業者においては、毎分約40枚の瓦が検査工程の製造ラインを通過する。従って、構築する画像処理システムにおいては毎分約40枚の処理速度が必要であり、許容される処理時間は瓦1枚当たり約1.5秒となる。産業応用を前提とする場合、前述した処理時間内に処理を完了させることが処理速度に関するシステム要件となる。

2.2 画像処理システムの構成

瓦の良否判定機能の実現を目的として、瓦の良否判定機能を内包した画像処理システムを構築した。構築した画像処理システムは、画像処理のソフトウェアとSBCのハードウェアで構成される。ここで、画像処理システムの処理について図2に示す。図2に示した2重線の処理が瓦の良否判定機能に相当する。瓦の良否判定機能は、瓦の良否を判定するMLモデル（以下、瓦の良否判定MLモデルと称す）を内包する。

* 情報・ヒューマンアミニティ科 (** 現：電子・電気技術科，
*** 現：情報技術・デザイン科)

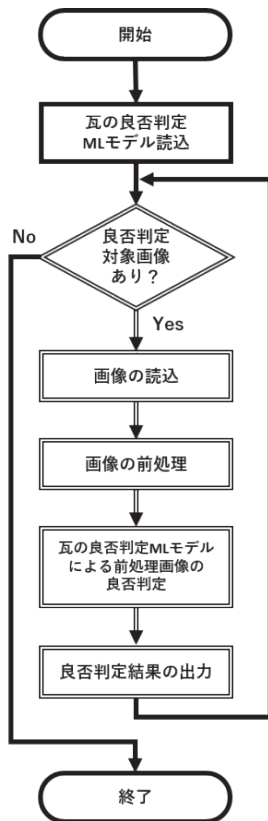


図2 画像処理システムの処理
(2重線の処理が瓦の良否判定機能)

表2 NVIDIA Jetson Nano の主要な仕様

項目	内容
GPU	128-Core NVIDIA Maxwell
CPU	Quad-Core ARM Cortex-A57 MPCore Processor
RAM	4GB 64bit LPDDR4
USB	USB 3.0 コネクタ および USB 2.0 コネクタ

図2に示した処理概要を説明する。まず瓦の良否判定 ML モデルを読み込む。読み込み完了後、良否判定対象画像が存在する場合には、良否判定対象画像の読み込みおよび前処理を実施する。その後瓦の良否判定 ML モデルによって良否判定を実施し、良否判定の結果を出力する。良否判定の結果を出力後、良否判定対象画像の存在有無の判定処理まで戻り、良否判定対象画像が無ければ処理を終了する。

SBC は NVIDIA Jetson Nano を利用した。NVIDIA Jetson Nano における主要な仕様について表2に示す¹⁾。NVIDIA Jetson Nano は、ハードウェアとしてグラフィックスプロセッシングユニット (GPU) を搭載しており、リアルタイムの画像処理に適している。

2.3 瓦の良否判定機能の構築および評価方法

図2で示した瓦の良否判定機能の主要な機能として、瓦の良否判定 ML モデルを利用した良否判定処理が挙げられる。瓦の良否判定 ML モデルにおける良否判定の性能が瓦の良否判定機能の性能に直結することから、瓦の良否判定 ML モデルの性能評価を以って瓦の良否判定機能の性能を

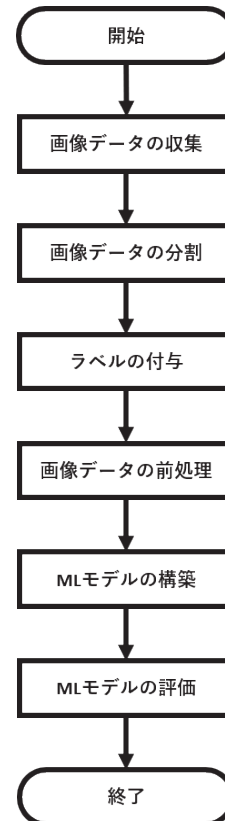


図3 ML モデル構築および評価の処理手順

評価する。以下、評価対象となる瓦の良否判定 ML モデルの構築および評価手順について説明する。

2.3.1 瓦の良否判定 ML モデルの構築と評価

ML モデルの構築および評価における処理手順を図3に示す。図3における各処理について述べる。

2.3.1.1 画像データの収集

瓦の良否判定 ML モデルは、ディープラーニングを利用して構築する。ディープラーニングは、大規模データからデータに含まれる特徴を ML モデルが学習する手法である。従って瓦の良否判定 ML モデルを構築するためには、大規模データとして画像データを収集する必要がある。

瓦の良否判定 ML モデルの構築において利用する画像データは、石州瓦の製造・販売を行う株式会社丸惣において撮影・収集した。収集した画像データは、良品画像が370枚、不良品画像が134枚であった。

2.3.1.2 画像データの分割

ML においては、収集したデータを訓練データ、検証データ、テストデータの3つに分割し、ML モデルの訓練、検証、テストに利用する。

瓦の良否判定 ML モデルの構築過程においては、瓦の良品画像370枚について、訓練データ、検証データ、テストデータとして、それぞれ220枚、75枚、75枚に分割した。また瓦の不良品画像134枚について、訓練データ、検証データ、テストデータとして、それぞれ84枚、25枚、25枚に分割した。以上のデータ分割の結果について表3に示す。

2.3.1.3 ラベルの付与

瓦の良否判定 ML モデルは、ディープラーニングに加えて教師あり学習を利用して構築する。教師あり学習は ML における学習手法の 1 つである。例えば特定の画像を分類しようとする場合、各画像データに対してラベルを付与し、画像データとラベルを利用することで ML モデルを構築する。

瓦の良否判定 ML モデルの構築過程においては、表 3 で示した良品画像と不良品画像に対して、それぞれ 0 と 1 のラベルを付与した。

2.3.1.4 画像データの前処理

前処理は、ML モデル訓練時における学習効率の向上や、ML モデル評価時における判定性能の向上を目的として実施する。

瓦の良否判定 ML モデル構築前に行う前処理手順について図 4 に示す。図 2 における「画像の前処理」の処理も、図 4 と同じ手順で行われる。図 4 に示すとおり、トリミング、グレースケール化、ランダム回転、リサイズを実施する。前処理の結果として、画像の幅および高さが 224 × 224 ピクセルとなるグレースケール画像が出力される。画像に対する前処理実施例として、図 1 に示した瓦の良品画像に対して前処理を実施した画像を図 5 に示す。

2.3.1.5 ML モデルの構築

ディープラーニングおよび教師あり学習を利用した一般的な ML モデルの構築方法として、訓練データ、検証データ、ラベルを利用して ML モデルの訓練および検証を実施する。まず画像分類を目的とした任意の ML モデルを選定する。次に訓練データとラベルを利用して、選定した ML モデルが保持するパラメータを更新する。そして検証データを利用して ML モデルが保持するパラメータ更新が適切かを検証する。以上の訓練および検証を繰り返し実施することで、ML モデルは訓練データおよび検証データを正しく分類することが可能となる。

瓦の良否判定 ML モデルの構築においては、表 3 で示した訓練データおよび検証データについて、図 4 で示した前処理を実施し、前処理実施後の訓練および検証データを利用した。ラベルは 2.3.1.3 で付与したラベルを利用した。以上の訓練データ、検証データ、ラベルを利用し、前述した ML モデルの構築方法を基に瓦の良否判定 ML モデルの訓練および検証を実施した。

また画像分類を目的とした ML モデルは様々なモデルが存在するが、瓦の良否判定 ML モデルにおいては MobileNetV2 の画像分類モデルを一部変更して利用した²⁾。具体的な変更内容として、瓦の良否判定 ML モデルは画像を良品または不良品の 2 つに分類するため、MobileNetV2 における分類数を 2 つに変更した点が挙げられる。

表 3 画像データ分割結果

項目	訓練 データ数 [枚]	検証 データ数 [枚]	テスト データ数 [枚]	合計 [枚]
良品画像	220	75	75	370
不良品画像	84	25	25	134

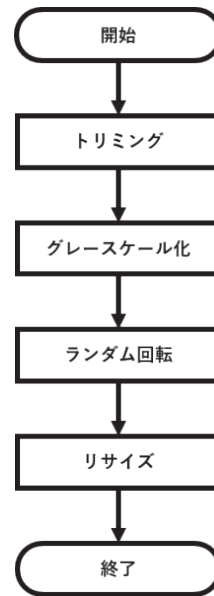


図 4 前処理手順

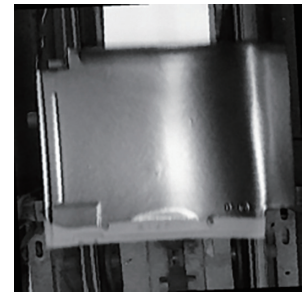


図 5 良品画像の前処理実施例

2.3.1.6 ML モデルの評価

ML モデルの構築後、ML モデルの性能評価を実施する。テストデータを利用して出力した ML モデルの判定結果とラベルが一致する割合を評価指標とする方法がある。この評価指標は一般的に正解率と呼ばれる。

瓦の良否判定 ML モデルの性能評価においては、表 3 で示したテストデータに対して、図 4 で示した前処理を実施して利用した。ラベルは 2.3.1.3 で付与したラベルを利用した。評価指標として正解率を利用した。

2.4 瓦の良否判定機能のシステム要件検証

2.1.3 で述べたシステム要件を満たすかどうかを検証する方法として、2.2 で述べた画像処理システムの構成で、瓦の良否判定機能の実行速度を計測する方法を用いた。実行速度の計測は、図 2 に示す 2 重線の処理を対象とした。具体的な計測方法として、図 2 における「良否判定対象画像あり？」の処理前の時刻と、図 2 における「良否判定結果の出力」の処理後の時刻を取得し、処理後と処理前の時刻の差を算出する方法を採用した。時刻の取得方法として、プログラミング言語である Python の Time モジュールを利用した³⁾。また実行速度の計測回数として、図 2 に示す 2 重線の処理を 100 回繰り返し実行して計測することとした。

図 2 における「瓦の良否判定 ML モデル読み込み」の処理は計測対象外とした。計測対象外とした理由として、画像処

理システムにおける瓦の良否判定 ML モデルの読み込みは、瓦の検査工程の製造ライン稼働前に実行が可能であり、システム要件の対象外となるためである。

本節で述べた計測方法および計測回数で実行速度を計測し、処理時間が1.5秒未満であるかを確認することで、瓦の良否判定機能がシステム要件を満たすかどうかを検証した。

3. 結 果

3.1 瓦の良否判定機能の構築および評価結果

2.3.1.6 で述べた ML モデルの評価について、評価結果を表4に示す。表4に示すとおり ML モデルにおける正解率は100%であり、構築した ML モデルは全テストデータに対して正しく良否判定できたことがわかる。従って構築した瓦の良否判定 ML モデルは、瓦の良否を適切に判定できることが確認できた。

3.2 瓦の良否判定機能のシステム要件検証結果

瓦の良否判定機能の実行速度計測結果を図6に示す。図6において、システム要件となる1.5秒を示すため、補助線として破線を付記している。図6より、実行速度計測1回目の速度は16.0秒であり、システム要件として許容される時間を上回っていることがわかる。一方で、実行速度計測2回目以降の速度はシステム要件を満たすことがわかる。実行速度計測1回目の実行速度が16.0秒となった理由として、速度計測時におけるSBCの処理が挙げられる。具体的には、MLモデルが判定の中間結果を格納するためのメモリ領域の確保等、SBCにおける初期化処理が実施されたためである。従って、SBCにおける初期化処理実施後という条件下であれば、瓦の良否判定機能はシステム要件を満たすことが確認できた。

3.3 まとめ

瓦の検査工程の自動化を目的として、筆者らは瓦の良否を判定する画像処理システムを構築した。構築した画像処理システムについて、瓦の良否判定機能の性能評価、および検査工程において許容される時間内に処理を完了させるというシステム要件を満たすかを検証した。結果として、構築した画像処理システムは瓦の良否を適切に判定し、かつSBCにおける初期化処理後という条件下であれば、システム要件を満たすことが確認できた。なおSBCにおける初期化処理は、製造ライン稼働前に実行し完了させておくことが可能であるため、製造ライン稼働中にはシステム要件を満たして運用することができる。

今後の課題として、表1の機能項目である瓦画像の撮影機能を構築し、2.1.3で述べたシステム要件を満たすかどうかを検証することが挙げられる。

表4 MLモデルの評価結果

項目	テストデータ数 [枚]	MLモデルの 正答数 [個]	正解率 [%]
良品画像	75	75	100
不良品画像	25	25	100

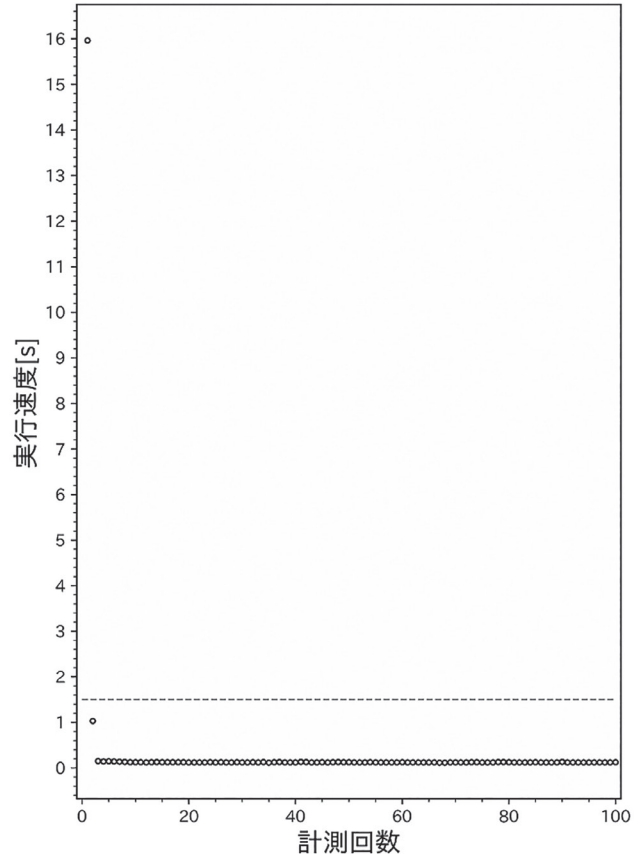


図6 瓦の良否判定機能の実行速度計測結果

謝 辞

本資料における瓦の良品画像および不良品画像は、株式会社丸惣からご提供いただいた。ここに記して謝意を表します。

文 献

- 1) NVIDIA Corporation. Jetson Nano 開発者キット. <https://www.nvidia.com/ja-jp/autonomous-machines/embedded-systems/jetson-nano-developer-kit/>. (参照 2023-7-13).
- 2) Mark, Sandler; Andrew, Howard; Menglong, Zhu; Andrey, Zhmoginov; Liang-Chieh, Chen. "MobileNetV2: Inverted Residuals and Linear Bottlenecks". IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, Utah, USA, June 18-23, 2018, IEEE, 2018, p.4510-4520.
- 3) Python Software Foundation. Time access and conversions. <https://docs.python.org/3/library/time.html>. (accessed 2023-7-13).