2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Towards intelligent robotic sole deburring: from burrs identification to path planning

Alessandra Tafuro[1], Luigi Cacciani[2], Andrea Maria Zanchettin[2] and Paolo Rocco[2]

Abstract— **Today, intelligent robotic manufacturing systems are reshaping the production industry. Using robots as actuators, multi-source sensors for perception, and Artificial Intelligence (AI) as decision-making systems, they can perform routine manufacturing tasks, surpassing the capabilities of traditional hard-programmed Computer Numerical Control (CNC) machinery. One specific challenge in footwear manufacturing is sole deburring, traditionally done manually by skilled workers. This paper focuses on developing a robust path-planning pipeline, comprising vision-based and Learning from Demonstrations (LfD) modules for autonomous deburring of soles. The vision-based module exploits Deep Learning (DL) techniques to handle key challenges such as precise segmentation of different soles types across diverse scenarios despite potential occlusions. Additionally, a novel method for burrs identification has been developed leveraging image processing and optimization techniques. Determining the optimal cutting tool orientation during sole deburring relies on human experience. The LfD module aims to impart this knowledge to the robot from videos of expert demonstrations, requiring adaptability to every new incoming sole that needs deburring. Experimental results showcase the method's performance and flexibility, underlining the potential to advance the field of the proposed approach.**

I. INTRODUCTION

Shoe manufacturing is a continuously evolving dynamic blend of expertise, technology, and innovation, which traditionally demands significant manual labor and dexterity, often in challenging work environments. Moreover, the landscape is evolving with the rise of small-scale customized and flexible production, aligning with modern customer preferences. In today's market, characterized by small batches and multiple variants, production lines must dynamically reconfigure process paths and manufacturing units [4]. When customization is extreme the current industrial solutions may fall short in addressing the needs of high-variability, high-quality, and low-volume production. The transition from *digital* to *intelligent* manufacturing is pivotal. In this domain, the sole deburring process emerges as a critical operation for ensuring the quality and safety of shoes. A *burr* is an unintended byproduct during manufacturing, manifesting as small projections extending beyond the designed workpiece surface and having a relatively small volume [5]. In the context of shoe sole production, burrs often arise from the injection molding process, specifically in the creation of outsoles. Expert operators utilize their visual analysis skills to enable precise removal of such defects, whereas automation with CNC machines facilitates exceptionally high production rates. However, conventional CNC robots offer speed and efficiency but lack the ability to adapt and change in response to required customization. The emergence of more intelligent robots seeks to bridge the gap between human flexibility skills and high-volume production, representing a significant shift. Additionally, the price of a comparable robotic solution for machining is typically 1/5-1/3 of the cost of a CNC machine [40]. The integration of new-generation AI, mainly machine and deep learning, plays a pivotal role in enhancing robotics across various sectors, offering solutions for autonomy and decision-making [7]. This paper delves into this transition, aiming to address the key challenges related to an efficient and automated robotic path planning pipeline for soles deburring. While studies on performing deburring with collaborative robots exist, as far as we know, this is the initial contribution concentrating specifically on shoe soles burrs removal. In particular, the main contribution of the paper is represented by a twofold path planning method made of:

- A vision-based module, composed of a blend of DL and conventional computer vision (CV) techniques for burr detection and tool position prediction.
- A second LfD module that generates the deburring tool orientation based on the output of the vision-based module. This module exploits videos of human experts conducting deburring tasks in an offline learning stage.

II. BACKGROUND KNOWLEDGE

A. Burrs identification

Currently, a diverse collection of methodologies exists for burrs detection, and the selection of the most suitable one is highly application-specific and dependent upon the workpiece's features, available instruments, and required burr quantities (height, thickness, volume, etc.) [5]. A possible categorization suggested by [10] is in *contact* (with touch sensors) and *non-contact* methods (laser, thermal, or vision-based). Touch sensors are the preferred choice when burrs lack clear visibility, which is not true for the examined case of burrs resulting from soles casting processes. Therefore, we focus here on non-contact methods. Several methodologies ([13], [15], [17]) employ laser displacement sensors (LDS) for direct burr size measurement: when mounted on the deburring robot, LDS allow to precisely measure the distance from the workpiece. Knowing all the other dimensions and the nominal geometry of the piece, the burr's height can be

[1]Alessandra Tafuro is with Politecnico di Milano, 20133 Milano, Italy and Istituto Italiano di Tecnologia, 16163 Genova, Italy (e-mail: alessandra.tafuro@polimi.it, alessandra.tafuro@iit.it)

[2] Luigi Cacciani, Andrea Maria Zanchettin and Paolo Rocco are with the Politecnico di Milano, 20133 Milano, Italy (e-mail: luigi.cacciani@mail.polimi.it, andreamaria.zanchettin@polimi.it, paolo.rocco@polimi.it).

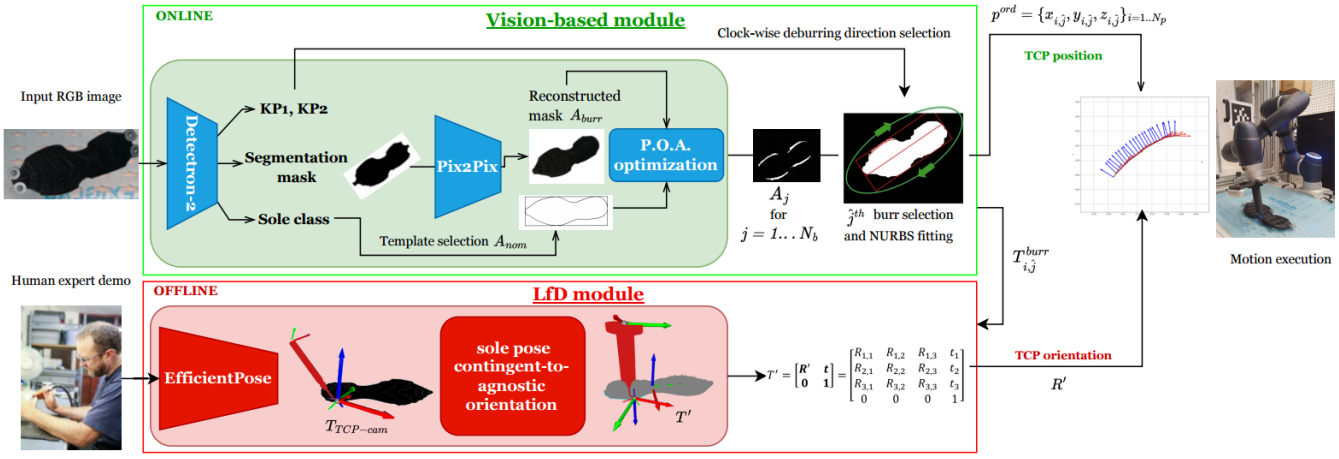

Fig. 1. Complete pipeline, including the vision-based module for burrs position identification and the LfD module for tool orientation prediction.

found point-wise. However, laser sensors are an expensive choice, and issues related to interference and diffraction tend to manifest [11]. [18] uses a high-temperature thermographic camera during steel slab cutting to detect burrs through thermal contrast. However, its applicability is limited to the specific process of steel slab cutting. In this context, vision systems coupled with image processing techniques represent a promising non-contact and cheaper solution. For instance, a 2D image of the workpiece can be transformed into binary form, enabling burr contour extraction and subsequent burr data derivation [11]. Thresholding can also be applied for detection exploiting the 3D scans of the object of interest ([19], [20]). Contour matching is another possible procedure: ideal geometric profiles are compared to identify peripheral defects [21]. The assumption behind this approach is that irregular burrs exhibit a pronounced variation in curvature, whereas a nominal profile appears smoother. Finally, a recent frontier comprises the use of DL for burrs detection ([22], [23], [25]). [23] uses a Convolutional Neural Network (CNN) to extract geometric contour features to be compared with a standard workpiece and detect the edges inconsistencies.

B. Path planning for deburring

Robotic deburring involves *motion planning* and *execution* [26]. The research focus of this work is the motion planning phase: exploiting the results of a defect identification strategy, the desired goal is to obtain the robot path. The subsequent motion execution stage translates these planned trajectories into physical deburring actions. Path planning for robotic deburring encompasses three primary approaches. (a) CAD/CAM-based approaches utilize Computer-Aided Manufacturing (CAM) software with Computer-Aided Design (CAD) models to select and plan deburring paths. (b) Sensor/Vision-based approaches employ specialized sensors or vision systems to detect workpiece geometry and define deburring paths. (c) LfD approaches involve guiding the robot through direct or indirect methods [12]. Literature showcases various path-planning methods integrating these three approaches. Among vision-assisted methodologies, [27] uses 2D cameras to identify workpiece features, serving as the basis for deburring paths. Innovative solutions leverage human expertise, like manual path drawing on the workpiece followed by digitalization through image processing for manually selecting key points on the workpiece [28]. Additional approaches involve offline teaching on a reference workpiece and online adaptation through 3D scans made by laser displacement sensors or alignment with the CAD model through CNNs [26].

III. PROPOSED APPROACH

The entire path planning pipeline (Fig. 1) depends on two inputs. One is the RGB image depicting the sole to be deburred from an overhead perspective. This is because soles casting methods tend to create burrs only along the lateral edges. The second input is a video of the expert performing the task. While the first input is essential for each new sole arrival requiring deburring, the second input is only necessary during the initial learning phase when the robot is acquiring the deburring skill. Two different modules compose the pipeline, one for each input:

- A **vision-based module**, made of a blend of DL and classic computer vision (CV) techniques, is employed for burr detection and tool position prediction. Whenever a new sole with burrs arrives, this module operates online to forecast the position of the robot's tool center point (TCP).
- A **LfD module** is employed to impose the deburring tool's orientation. This module needs to be executed only one time during the initial skill learning stage. Subsequently, upon the arrival of a new sole with burrs, it generates the tool orientation based on the output of the vision-based module.

IV. VISION-BASED MODULE

In our view, the vision-based module must provide the burrs' positions, irrespective of the sole type (which often varies in industrial settings), lighting conditions, occlusions or placement of the sole in the workspace. Additionally, upon identifying a burr, it is essential to establish local reference frames on it to express the tool's orientation uniquely. Therefore, the current section is subdivided according to the sub-goals of the vision-based module, which can be listed

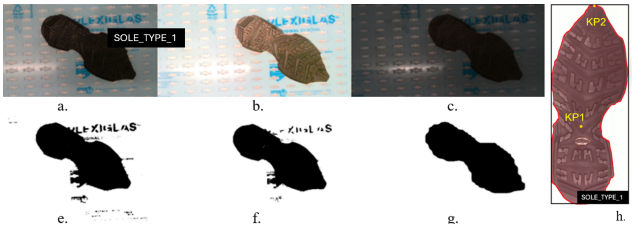

Fig. 2. **a.** Sample of RGB image, input to the pipeline. **b. c.** Same sample under two modified lighting conditions. **e. f. g.** Segmentation masks resulting from *simple*, *adaptive local thresholding* and *Detectron-2*. **h.** Detectron-2 outputs : Sole class, *KPs* (*center of the sole KP1* and the *sole tip KP2*), bounding box (black) and segmentation mask (red).

as: (1) soles classification and segmentation, (2) occluded profile reconstruction, (3) burrs identification and (4) TCP trajectory generation.

1) Soles classification and segmentation: For extracting the profile of the sole with burrs, the simplest approach involves classic computer vision (CV) image processing techniques. We tested *simple*, *Otsu's global thresholding* [30] and *adaptive (local) thresholding* [31]. However, their lack of robustness against varying lighting conditions (Fig. 2) prompted us to move to a deep learning (DL) approach. DL methods offer also capabilities for classification (distinguish between different types of soles) and key points (KP) prediction. The most representative approach according to [9] is Detectron-2 [1] launched by Facebook, a state-of-the-art DL framework for object detection, built upon Mask Regional CNNs [29]. The outputs of our custom Detectron-2 model are the bounding box (BB), the segmentation mask, and KP (namely the *center of the sole* KP1 and the *sole tip* KP2) of the sole (Fig. 2). Detectron-2 achieves exceptional accuracy across various benchmarks, surpassing many pre-trained architectures. It is designed for high efficiency and scalability, making it one of the faster frameworks for both training and inference. Additionally, it benefits from a robust and active community, with ongoing support from Facebook AI, and is built on PyTorch [41], which allows for easy extension for specific downstream tasks.

2) Occluded profile reconstruction: During the deburring process, the sole is subject to a certain level of force, so securing it in place above a fixed base is imperative. An appropriate approach is to grasp the sole, but any gripping system inevitably obstructs a portion of the sole's surface. The presence of such profile occlusions could potentially affect the accuracy of the burr identification method, even if the clamps are positioned over parts of the profile with no burrs (as described in Sec. VI). This is because the method relies on overlaying the desired sole profile onto the one with burrs (Sec. IV-.3). The detection of this latter profile utilizes the methods outlined in the preceding subsection and any alterations resulting from occlusions might lead to an inaccurate alignment of the two profiles. Inspired by [35], to reconstruct the occluded portions, a Conditional Generative Adversarial Network (cGAN) was exploited, specifically Pix2Pix, a framework for image-to-image translation developed by Isola et al. in 2016 [2]. The cGAN architecture was selected for its proven success in tasks such as image-to-image translation and structured image reconstruction. More recent models like Masked Autoencoders (MAEs) [36] were not chosen, as they are primarily designed for representation learning rather than complex reconstruction tasks. Additionally, the adversarial training mechanism enhances the generation of more realistic and higher-quality outputs, making it a more suitable choice for this application.

3) Burrs identification: The ultimate objective is to eliminate burrs, so a novel burr identification method was developed leveraging *CV* and optimization techniques. The sole can vary in size, be oriented, and be placed randomly within the acquired image, and the proposed method remains effective. The method relies on the correct overlay of a desired sole profile (template) onto the burr-containing sole contour, so that the area in between them represents the material to be removed. The templates of the different sole types can be obtained a priori by processing images of the deburred soles with a white background (through simple global thresholding and contour detection functions from classic CV). To obtain the templates, the sole contour scale and position are standardized: the bounding box BB_{nom} containing the desired sole contour C_{nom} is scaled to have a unitary longest side, and its center is positioned at the middle of the image (wxh).

$$BB_{nom} = [u_{nom}, v_{nom}] = [w/2, h/2] \tag{1}$$

When used online, the vision-based module takes the top-view RGB image and extracts the contour of the sole with burrs C_{burr} (exploiting Detectron-2 segmentation and Pix2Pix reconstruction). Its oriented bounding box BB_{burr} is then calculated. Relying on the sole class predicted by Detectron-2, also the corresponding C_{nom} and BB_{nom} are selected as templates. Once $BB_{burr}, C_{burr}, BB_{nom}$ and C_{nom} are available, an optimization process takes place (outlined in Fig. 3 and Alg. 1) to maximize the contours matching. The objective is to find the optimal *scale* Δh, *rotation* $\Delta\gamma$, and *translations* Δx and Δy of the nominal template in pixel space, so that the area in between C_{nom} and C_{burr} represents only the burrs. We propose a metric to be maximized called *Percentage of Overlapping Area* ($P.O.A. \in [0, 100]$) defined as:

$$P.O.A. = (A_{burr} \cap A_{nom})/A_{nom} \tag{2}$$

where A_{burr} and A_{nom} are the sets of pixels inside the 2 contours. Ideally, when the nominal template is translated, scaled and rotated so that $P.O.A = 100\%$, we reach our goal and $A_{burr} \oplus A_{nom}$ represents exactly the burrs in pixel space. As A_{nom} is function of Δh, $\Delta\gamma$, Δx and Δy, the optimization problem we want to solve can be formulated as:

$$\Delta h^*, \Delta\gamma^*, \Delta x^*, \Delta y^* = \underset{\Delta h, \Delta\gamma, \Delta x, \Delta y}{\operatorname{argmax}} \; P.O.A. \tag{2}$$

The optimization relies on a direct search of the solution, and the starting point is obtained aligning the centers of BB_{nom} and BB_{burr} and their orientations $\alpha_{BB_{nom}}$, $\alpha_{BB_{burr}}$.

$$\left\{ \begin{array}{c} BB_{nom} = [u_{nom}, v_{nom}] = BB_{burr} = [u_{burr}, v_{burr}] \\ \alpha_{BB_{nom}} = \alpha_{BB_{burr}} \end{array} \right\} \tag{3}$$

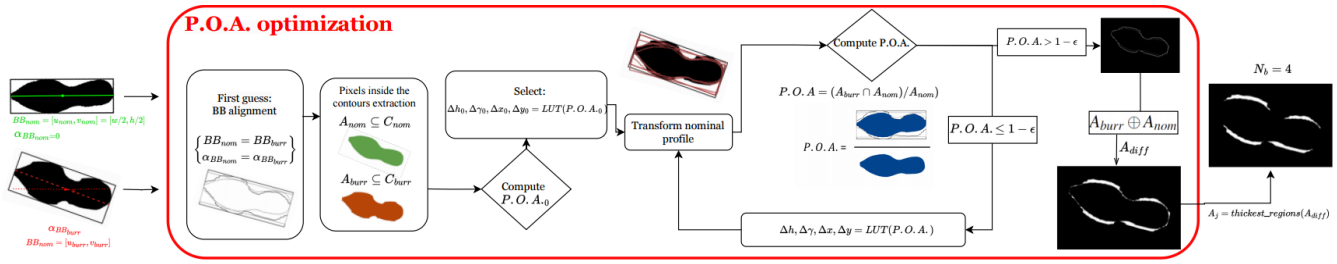

Fig. 3. Burrs identification algorithm through *P.O.A.* metric optimization.

At this initial stage, since the transformations to the nominal profile are based on data obtained from BB_{burr} (minimum-area rectangle around the contour), they are significantly influenced by possible uneven distribution of burrs. So, the method proceeds with subsequent relocation and scaling of C_{nom}, calculating the new *P.O.A.* at each iteration. To enhance the convergence speed, the search space for the optimization variables at every iteration depends on the *P.O.A.* current value. When the overlap is low, broader step sizes speed up the search; instead, as the *P.O.A.* increases, precision becomes more crucial, so reduced step sizes allow for finer refinement at pixel level. This adaptive approach balances the search space exploration, accelerating the convergence of the process compared to a straightforward global search method.

This is implemented with a lookup table LUT:

$$\Delta h, \Delta\gamma, \Delta x, \Delta y = LUT(P.O.A.) \tag{4}$$

It is to be noted that *P.O.A.* uses A_{nom} at the denominator, so there is the risk that Δh is largely diminished so that A_{nom} is entirely included in A_{burr}. To address this, the algorithm starts with the maximum scale and downscales it iteratively, yielding the optimal overlapping solution with the largest scale possible. Considering ε as a user-defined tolerance, the process stops as soon as $P.O.A. > 1-\varepsilon$. Finally, a pixel-wise XOR ($A_{burr} \oplus A_{nom}$) logical operator is used to obtain the segmentation of the burrs. Subsequent filtering retains only the thickest portions, obtaining N_b burrs binary masks A_j for $j = 1...N_b$.

4) TCP trajectory generation: Each identified burr is selected subsequently for removal. The deburring path in the plane is determined by intersecting the area of the burr $A_{\hat{j}}$ with the sole desired one A_{nom} in pixel space ($A_{\hat{j}} \cap A_{nom}$), obtaining a collection of N_p deburring points $O'_{px} = \{u_{i,\hat{j}}, v_{i,\hat{j}}\}_{i=1..N_p}$.

These O'_{px} are considered as control points for the interpolation of a Non-Uniform Rational Basis-Splines (NURBS) curve of degree 2, with unitary weights. The O'_{px} coordinates are then transformed from pixels to the camera frame ($p^c = \{x^c_{i,\hat{j}}, y^c_{i,\hat{j}}, z^c_{i,\hat{j}}\}_{i=1..N_p}$) exploiting the intrinsic camera parameters and the known vertical distance of the camera from the sole z_c. From the camera frame, the coordinates are transformed to the robot base frame through the transformation matrix T_{cam} describing the camera pose w.r.t. the robot base frame (camera's extrinsic parameters), obtaining

Algorithm 1 Percentage of Overlapping Area maximization.

Input: C_{burr}, BB_{burr}, C_{nom}, BB_{nom}, LUT($P.O.A. \rightarrow \Delta h, \Delta\gamma, \Delta x, \Delta y$)
Outputs: N_b binary masks A_j for $j = 1...N_b$ representing burrs.

Initial transformation of the nominal profile:
$BB_{nom} = [u_{nom}, v_{nom}] = BB_{burr} = [u_{burr}, v_{burr}]$
$\alpha_{BB_{nom}} = \alpha_{BB_{burr}}$
Pixels inside the contours extraction: $A_{burr} \subseteq C_{burr}$, $A_{nom} \subseteq C_{nom}$
P.O.A. computation: $P.O.A._0 = (A_{burr} \cap A_{nom})/A_{nom}$
$i = 0$
Rotation, translation and scale steps selection from lookup table:
$\Delta h_0, \Delta\gamma_0, \Delta x_0, \Delta y_0 = LUT(P.O.A._0)$
while $P.O.A. \leq 1-\varepsilon$ **do**
    Transformation of the nominal profile using $\Delta h_j, \Delta\gamma_j, \Delta x_j, \Delta y_j$
    Pixels inside the template extraction: $A_{nom} \subseteq C_{nom}$
    P.O.A. computation: $P.O.A._j = (A_{burr} \cap A_{nom})/A_{nom}$
    Rotation, translation and scale steps selection from lookup table:
    $\Delta h, \Delta\gamma, \Delta x, \Delta y = LUT(P.O.A.)$
    $i = i + 1$
end while
Pixel-wise XOR logical operator: $A_{diff} = A_{burr} \oplus A_{nom}$
Thickest portions filtering: $A_j = thickest\_regions(A_{diff})$ for $j = 1...N_b$

Algorithm 2 Clockwise deburring direction imposition.

Input: $A_{burr}, A_{\hat{j}}, p = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$
Outputs: Ordered set of deburring points $p^{ord} = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$

$BB_{burr} = minAreaRect(A_{burr})$, $L = CenterLine(BB_{burr}) = f(x,y)$
$C_u, C_v = Center(minEnclosingCircle(A_{\hat{j}}))$
$var_x, var_y = Variance([\{x_{i,\hat{j}}\}_{i=1...N_P}, \{y_{i,\hat{j}}\}_{i=1...N_P}])$
is_above = $f(C_u, C_v) < 0$
if *is_above* and $(var_x \geq var_y)$ **then**
    $p^{ord} = p(increasing\ x) = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$
else if *is_above* and $(var_y \geq var_x)$ **then**
    $p^{ord} = p(increasing\ y) = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$
end if
if not *is_above* and $(var_x \geq var_y)$ **then**
    $p^{ord} = p(decreasing\ x) = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$
else if not *is_above* and $(var_y \geq var_x)$ **then**
    $p^{ord} = p(decreasing\ y) = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$
end if

$p = \{x_{i,\hat{j}}, y_{i,\hat{j}}, z_{i,\hat{j}}\}_{i=1..N_p}$. Then, the points are ordered to allow for a clockwise execution of the path. The equation L of the center line along the longest edge of BB_{burr} is extracted and the selected burr is classified as either *"above"* or *"below"* it, exploiting the center coordinates (C_u, C_v) in pixel space of the smallest enclosing circle of $A_{\hat{j}}$. If C_v is less then the value obtained by substituting C_u into the equation of the center line L, the variable *is_above* = 1, otherwise *is_above* = 0. The points p are then reordered in p^{ord} to follow a clockwise direction, on the basis of the variances of the x and y coordinates (Alg 2). Finally, in every p, a local reference frame $T^{burr}_{i,\hat{j}}$ is uniquely determined. Normal n and

Fig. 4. **a.** Manual deburring tool and its 3D-printed support for robot installation. **b.** Local RFs $T^{burr}_{i,\hat{j}} = [t,n,z']$. **c.** *BlenderProc* generated image. **d.** *EfficientPose* prediction on one frame showing the operator executing the task. **e.** 3D reconstruction of the scene in *d.*

tangent t vectors to the NURBS curve are identified: since two normals can be computed, pointing outside or inside the sole profile, the one that points far from *KP1* (Detectron-2 prediction representing the center of the sole) is selected as n. t instead is chosen to rotate clockwise along the profile and z' points upwards with respect to the working plane. With these requirements, the N_p local reference frames on the $\hat{j}^{th}$ burr $[t,n,z']$ are always uniquely determined (Fig. 4 b.).

V. LfD MODULE

From the vision-based module it is possible to obtain the cutting tool path points, p^{ord}. Imposing also the optimal tool orientation is crucial, since the cutting blade is only present on one side of the tool's tip. There are basically two possible arrangements in robotic deburring: (1) the robot moves the workpiece, and the deburring tool is fixed; (2) the robot carries the deburring tool, and the workpiece is clamped in place. The latter is usually chosen since it offers more flexibility and accuracy [8]. Additionally, we are considering scenarios in which a non-actuated universal spare blade deburring tool is mounted on the robot flange through a custom holder (Fig. 4 a.). No spindle motor is used, motivated by the cost-effectiveness of the solution and inspired by [8]. Our study proposes teaching the robot the ideal tool orientation through indirect human demonstrations (simple videos of experts performing the task). Direct teaching methods, like kinesthetic teaching, were discarded to enhance the naturalness of the deburring gesture, crucial for correct task execution due to the tool's small cutting edge and required precise positioning. The LfD module would allow the system to learn from skilled human workers, capturing the nuanced craftsmanship required and translating it into automated processes that can adapt to varying designs, without the need of CAD models.

1) Cutting tool orientation prediction through LfD: To keep the needed hardware simple, we wanted to extract the 6D pose of the manual tool operated by a demonstrator from simple RGB images. We chose a DL approach, precisely EfficientPose [3], known in the literature for its exceptional accuracy of pose estimation. Indeed, according to [38], it is the most accurate RGB-based method evaluated on *LineMOD dataset* [14]. As obtaining accurate 6D pose annotations in numerous images can be time-consuming and costly, we generated data with the open-source framework for synthetic image rendering *BlenderProc* [34]. This enables the

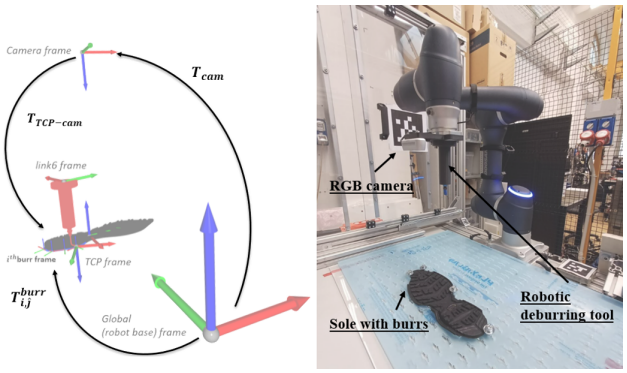

Fig. 5. Experimental setup and 3D representation of the different RFs.

placement of the 3D model of the manual tool in a known annotated pose (Fig. 4 c.).

2) Sole pose contingent-to-agnostic orientation: EfficientPose outputs $T_{TCP-cam}$ (the tool pose w.r.t. the camera RF) depending on the particular pose of the sole during the expert demonstration (Fig. 4 d.). For autonomous execution of the tasks, the TCP orientation must be transmitted in the robot base frame and should be adaptable to every new incoming sole. Hence, it is crucial for the orientation information to be extracted in a manner that is detached from the specific pose of the sole during demonstrations. Our proposed approach is to formulate the tool orientation in the $[t,n,z']$ local reference frames computed by the vision-based module (Fig. 4 e.).

$$T' = (T^{burr}_{i,\hat{j}})^{-1} T_{cam} T_{TCP-cam} \tag{6}$$

T' is estimated in each acquired frame of the expert demonstration video. The rotational part of matrix T' is extracted and expressed in quaternions to compute an average value over the frames. Expressing this average orientation as a rotation matrix, we obtain R'.

Finally, during online path prediction, the LfD module imposes the TCP orientation described by R', formulating it in robot base frame.

$$R_{TCP} = T^{burr,rot}_{i,\hat{j}} R' = \begin{bmatrix} I_3 & 0_{1x3} \end{bmatrix} T^{burr}_{i,\hat{j}} \begin{bmatrix} I_3 \\ 0_{1x3} \end{bmatrix} R' \tag{7}$$

VI. EXPERIMENTAL VALIDATION

The experimental setup proposed in this study comprises the sole requiring deburring clamped to the working plane

Fig. 6. Three different sole types used for experimental validation.

using circular rings positioned in the parts of the profile without burrs, tightened with screws, an *Intel RealSense RGB camera* attached to the end effector, and a *Doosan A0509* equipped with a custom deburring tool (Fig. 5).

A. Vision-based module

1) Soles classification and segmentation: Experiments with three backbone models from Detectron-2 model Zoo API were carried out, namely R50-FPN, R101-FPN and X101-FPN. A novel dataset was created to detect and segment shoe soles. 71 images were captured with a resolution of (640x480): 3 soles types (Fig. 6), with and without burrs, with variations in orientation, background, and clamping configuration, were included. The dataset was split into training (80%), validation (10%) and testing (10%) subsets. Table I summarizes the results in terms of segmentation Average Precision (AP) values over the three classes, with Intersection over Union (IoU) thresholds of 0.975 and 0.99. The results of KP detection are expressed in terms of AP at the same IoU thresholds, but at each IoU threshold, we take the average results from 0.5, 0.3, and 0.1 OKS. OKS [1] is the standard performance metric used by Detectron-2 and MSCOCO [32] for KP detection. Despite all models showing excellent segmentation and KP detection AP, the X101-FPN one was chosen due to the best performance under the most stringent demand.

2) Occluded profile reconstruction: The training dataset for Pix2Pix comprised pairs of images: a ground truth image representing the unobstructed sole segmentation binary mask and an image with randomly created occlusions on the sole profile (Fig. 7). 25 ground truths were created introducing random rotations and translations on 10 different sole binary masks. For each ground truth, 40 images with random occlusions were generated drawing white circles (of radius from 10 to 30 pixels) at random along the black mask contour, to simulate the effect of ring and screw occlusions. The final dataset comprises 10000 elements (80% training, 20% validation). The model was evaluated on a test set of 20 new binary masks of soles with actual ring and screw occlusions after Detectron-2 detection. The performance on the test set (AP@IoU0.9 = 89.87%) shows that the trained network is able to reconstruct the profiles (Fig. 7). It is important to highlight the necessity of an extra stage in refining the binary mask produced by Detectron-2, before allowing Pix2Pix to reconstruct the occluded profile. This step involves smoothing outlines around segmented soles by filling isolated white pixels through binary dilation. Without this refinement, Pix2Pix may misinterpret irregularities as minor obstructions, leading to a blurred reconstructed profile.

TABLE I

DETECTRON-2 SEGMENTATION AND KP PREDICTION RESULTS.

	AP Segm.		AP KP	
Backbone	IoU@0.975	IoU@0.99	IoU@0.975	IoU@0.99
R50-FPN	91.7%	73.4%	90.5%	71.4%
R101-FPN	**100%**	71.6%	97.4%	70.5%
X101-FPN	98.9%	**77.5%**	**98.2%**	**76.5%**

TABLE II

SEGMENTATION PERFORMANCES: DETECTRON-2 VS CLASSIC CV

IoU (0% - 100%) quantifies ground truth and predicted segmentation masks overlap.

Model	Standard	Brighter	Darker
Simple Global thres.	**97.86%**	2.90%	30.35%
Otsu's global thres.	94.82%	94.94%	92.99%
Adaptive thres.	95.88%	95.78%	94.50%
Detectron-2	96.69%	**96.10%**	**96.65%**

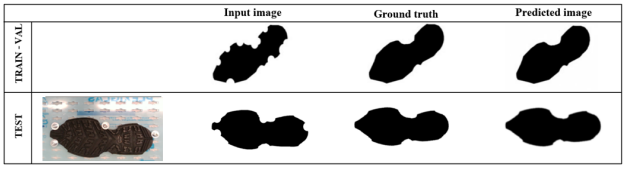

Fig. 7. *Pix2Pix* train/validation and test sets samples.

B. LfD module

1) Cutting tool orientation prediction through LfD: The dataset used to train EfficientPose comprises 10000 scenes generated with *BlenderProc*, split into 80% training, 10% validation and 10% test sets. To address the reality gap, physical features like surface roughness and a varying point light source were added and the chosen backgrounds showed soles and hands. The model was trained for 175 epochs with a reduction of 50% in the learning rate after 25 epochs of stagnation. The initial and minimum learning rates were 10^{-4} and 10^{-7}, respectively. The scaling factor ϕ was set to 0 since there was no need to oversize the network (there's only one object pose to be recognized) at the expense of the computational cost. In the case of the deburring tool, which features a handle with rotational symmetry, the matching between points can become ambiguous; so, the standard ADD [16] metric used for 6D pose evaluation was replaced by the ADD(-S) [16] metric. The model achieved an ADD(-S) value of 86.85% on the test set.

2) Sole pose contingent-to-agnostic orientation: For tool orientation learning purposes, 3 videos of deburring operations made by an expert were recorded, and 116 frames from each video were analyzed. The 348 tool poses outputted by EfficientPose are manipulated to extract R'.

C. Complete path planning accuracy

The final validation attempts to assess the accuracy of the complete pipeline by comparing its path prediction with the one demonstrated by an expert who has physically manipulated the robot (Fig. 8). 5 tests were conducted with 3 different soles positioned randomly on the working plane. For every sole, the deburring paths of all the detected burrs were considered for a total of 14 predicted and demonstrated

paths. The Robot Operating System (ROS) [39] was employed for communication purposes. The robot moves along a spline curve path, connecting the list of provided points p^{ord} in the task space. Results are shown in Table III. The average Euclidean distance between predicted and operator-collected trajectories is of 4.16 mm. The result is deemed highly acceptable for the study's objective, emphasizing the need for a control algorithm to address inherent path planning uncertainties and ensure the continuous contact with the material and adequate cutting force.

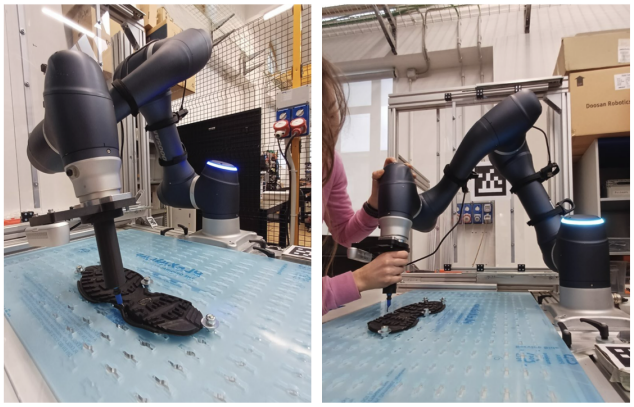

Fig. 8. Final validation to assess the path pipeline accuracy by comparing its prediction with the one demonstrated by an expert.

TABLE III

ERROR BETWEEN PREDICTED AND DEMONSTRATED PATHS IN [MM]

Burr	1	2	3	4	**Av. Euc. Dis.**
Sole 1	6.2	5.3	4.2	3.2	
Sole 2	1.6	-	-	-	
Sole 3	5.6	4.9	-	-	**4.16**
Sole 4	6.7	7.5	7.5	5.5	
Sole 5	3.2	1.2	3.2	-	

VII. CONCLUSIONS

As manufacturing evolves into the intelligent era, this research aims to advance autonomous robotic deburring in the footwear industry. As far as we know, this is the first contribution to employ a collaborative robot for shoe soles deburring. We propose a path planning pipeline comprising a vision-based module and a LfD module. The vision-based module handles sole recognition, occlusions reconstruction and burrs identification. The optimal orientation of the tool is learned from RGB videos of expert demonstrations, exploiting the LfD module. The experiments demonstrated the capability of the pipeline to identify defects and predict the relative path accordingly. Future work will be focused on motion execution and the actual cutting stage.

REFERENCES

[1] Yuxin Wu et al., 2019, Detectron-2, https://github.com/facebookresearch/detectron2

[2] P. Isola et al., Image-to-image translation with conditional adversarial networks, CVPR, 5967–5976, 2017.

[3] Y. Bukschat et al., Efficientpose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach, APIN, 2020.

[4] B. Chen et al., Smart factory of industry 4.0: Key technologies, application case, and challenges, pp:1–1, 12, 2017.

[5] J. Aurich et al., Burrs analysis, control and removal, CIRP Ann, 519–542, 2009.

[6] B. Wanget al., Smart manufacturing and intelligent manufacturing: A comparative review. Engineering, 7(6):738–757, 2021.

[7] L. Alatabani et al., Machine Learning and Deep Learning Approaches for Robotics Applications, pp:303–333, 05-2023.

[8] M. E. Matour et al., Force Controlled Deburring using a Collaborative Robot, MMAR, 2022.

[9] Wan Xing et al., A review on object detection algorithms based deep learning methods, JEESR, 2023.

[10] E. Bahce et al., Burr measurement method based on burr surface area, IJPEM-Green Technology, 2020.

[11] K. C. Lee et al. Burr detection by using vision image, Int. J. Adv. Manuf. Technol., 8:275–284, 1993.

[12] A. Correia et al., A Survey of Demonstration Learning, arXiv, 2303.11191, 2023.

[13] K. Shimokura and S. Liu, Programming deburring robots based on human demonstration with direct burr size measurement, ICRA, 1994.

[14] Hinterstoisser et al.,Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes, ICCV, 858–865, 2011.

[15] B.-H. Kang et al. Robot intelligence and flexibility for smooth finishing of car body. IMS, 1997.

[16] S. Hinterstoisser et al. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes, ACCV, 2012.

[17] Y. D. Lee et al. Robotic deburring strategy using burr shape recognition, IROS, 1999.

[18] Wulf C et al., Automatic Residual Burr Detection on Steel Slabs by a Thermographic System. International MPT, 2007.

[19] Y. Nakao aet al, Measurements and evaluations of drilling burr profile, Journal of Engineering Manufacture, 2006.

[20] X. Chenet al., Design of burr detection based on image processing. Journal of Physics: Conference Series, 2019.

[21] D.M. TSAI and W.-J. Lu. Detecting and locating burrs of industrial parts. IJPR, 34(11):3187–3205, 1996.

[22] R. M R and S. Chiddarwar. A causality-inspired data augmentation approach to cross-domain burr detection using randomly weighted shallow networks, IJMLC, 2023.

[23] Lin, X. et al. Intelligent detection of edge inconsistency for mechanical workpiece by machine vision with deep learning and variable geometry model, Appl Intell 50, 2105–2119, 2020.

[24] Bradski et al., The OpenCV Library, Dr. Dobb's Journal of Software Tools, 2000.

[25] A. P. Rifai et al., Evaluation of turned and milled surfaces roughness using convolutional neural network, Measurement, Volume 161, 2020.

[26] I. F. Onstein et al., Deburring using robot manipulators: A review. SIMS, 2020.

[27] Z. Lai et al. Integration of visual information and robot offline programming system for improving automatic deburring process, ROBIO, 2018.

[28] B. Solvang et al. Vision based robot programming. 949– 954, 05-2008.

[29] Kaiming He et al., Mask R-CNN,1703.06870, arXiv, 2017.

[30] N. Otsu. A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics, 9(1):62–66, 1979.

[31] T. Romen Singh et al., A New Local Adaptive Thresholding Technique in Binarization, 1201.5227, arXiv, 2012.

[32] T.Y. Lin et al., Microsoft coco: Common objects in context, ECCV, 2015.

[33] https://www.ati-ia.com/products/deburr/deburring_cdb_main.aspx

[34] Maximilian D.et al., BlenderProc2: A Procedural Pipeline for Photorealistic Rendering, J. Open Source Softw., 2023.

[35] Kaziwa Saleh et al.Generative Adversarial Network for Overcoming Occlusion in Images: A Survey, Algorithms, 2023.

[36] Kaiming et al. Masked Autoencoders Are Scalable Vision Learners, CVF, 2021.

[37] P. Virtanen et al., SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. 202, Nature Methods, 17(3):261-272, 2019.

[38] Jason Zhaoxin et al., Deep Learning on Monocular Object Pose Detection and Tracking: A Comprehensive Overview, ACM Computing Surveys 55:1-40, 2021.

[39] SAIL, Robotic operating system. URL: https://www.ros.org.

[40] A. Brunete et al. Hard material small-batch industrial machining robot, ROBOT CIM-INT MANUF, vol. 54, pp. 185–199, 2017.

[41] Paszke et al. Automatic differentiation in PyTorch, NIPS-W, 2017.