Feedbot: A Vision Augmented Feeding Robot

Alexandre Candeias

***Abstract*—People with disabilities are unable to execute several simple daily tasks, such as feeding or dressing, and often rely on other people to help them. Researchers have over time developed robotic feeding assistants to help at meals so that people with disabilities can live more autonomously. However, current commercial feeding assistant robots acquire food without feedback on acquisition success and deliver it in a preprogrammed location.**

We propose Feedbot, an autonomous feeding robot arm that is augmented with vision to be capable of adapting his trajectories in real-time to the user and of having visual feedback on the food acquisition. Feedbot was tested in a real meal scenario and using two different robot arms.

We show how Discriminative Optimization (DO) can be applied to track the user's head pose so that the food can be effectively brought all the way to the user's mouth, rather than to a preprogrammed feeding location. Our results show the ability of DO to track the head pose, while achieving state of the art performance. Finally, we show that visual feedback can improve the effectiveness of food acquisition.

***Index Terms*—Assistive Technologies, Manipulation Aids, Feeding Assistance, Computer Vision, Head Pose Tracking.**

I. Introduction

Some people that live with strong disabilities are incapable of doing their Activities of Daily Living (ADLs). These activities are simple self-care tasks such as feeding, bathing, dressing or shaving. Since these people are not able of doing these tasks by themselves, they need help from other people like caregivers or nurses. Helping people in their ADLs can be a difficult and tedious task to the caregiver. People living with disabilities are also less likely to live autonomously.

Robots can play a significant role in helping society to deal with these two problems. On one hand, they can provide more autonomy to people who suffer from disabilities. On the other hand, robots will free caregivers of certain "mechanical" tasks so that they can concentrate on the relational role.

In this work, we will focus on the task of autonomously feeding a user. Feeding is an essential Activity of Daily Living (ADL) and was reported by the users to be one of the most difficult tasks to execute controlling a joystick [1]. Feeding involves bringing the food from the plate location to the user's mouth. Even though this can be thought as being a simple task, it is not and it can be even more challenging for a robot, since it requires a perception of the environment and of the user's pose. The perception of the environment is required because the robot must be capable of detecting where the food is located and to scoop the food from the plate. At the same time, the robot must also be capable of detecting changes in the environment, for example, if there is food present on the plate. The perception of the user's pose is also important since the robot should move the end-effector to the mouth position and adapt its trajectories to different users that may have involuntary and diverse amplitudes of movements.

Alexandre Candeias, Instituto Superior Técnico - Institute for Systems and Robotics , Lisboa, Portugal, alexandre.candeias@tecnico.ulisboa.pt.

In this work, we propose the introduction of a vision system to achieve a full autonomous feeding robot that is capable of adapting his trajectories to the user's movements and of having feedback on the food acquisition.

The major contributions of this work are:

1) **Autonomous Feeding Robot (Feedbot).** We built a fully autonomous robotic system that is capable of:
   a) Acquiring food from a static plate using a demonstration of a scooping trajectory;
   b) Detecting if the acquisition of the food was successful;
   c) Real-time tracking of the user's mouth position;
   d) Real-time adapting the trajectory of the robot end-effector from the plate to the user's mouth.
2) **RGB-D head pose tracking system.** We propose a tracking system with the ability of tracking the user's head at 28FPS and has the same accuracy as state of the art methods. This system makes it possible to track the user's mouth with a Mean Absolute Error (MAE) below 1cm. Furthermore, our tracking system is very robust to extreme head pose variations being capable of detecting the correct head pose even in the case where the head pose is in profile.
3) **RGB based food acquisition feedback system.** Using a RGB camera we developed a food acquisition feedback system that is capable of detecting if there is food on the spoon or how much food is present. Real experiments show that this system provides significant improvements in terms of the food delivered to the user per distance traveled.

Feedbot allows to feed different users with different types of food. Moreover, our system is fully independent of the robot arm platform. We tested our system in two different robot arms, namely, Kinova MICO and Nyrio One [2].

II. State of the Art

A. Assistive Feeding Robots

An assistive robot is a robot that provides aid or support to the human user [3], for example, robots that help the humans on ADLs. Researchers have made efforts in this type of assistive robots, making possible to use robots in some of these tasks like dressing [4][5], shaving [6] and drinking [7].

One of the most important ADLs and the focus of this work is the task of feeding. The first effort to bring a fully capable robot arm to help people with upper arm disabilities was Handy1 [8]. Handy1 was specially built for people with cerebral palsy and it was tested with more than 50 users. With this robot, users could choose between 7 different food types

using input buttons and the robot could move the food all the way from the plate to a preset mouth location. Although this robot was widely used and helpful, it could only move to a predefined location and it could not be used by people who cannot control the input buttons.

To overcome the problem of controlling the robot by this kind of users, in [9] the authors propose a new robot where the control is done by moving a pointing laser with the head. Following the developments in research, feeding assistant robots like Obi [10], MySpoon [11], MealBuddy [12], iEAT [13] and Bestic [14] became available for commercialization. All those platforms have some of the features present in Handy1[8] and [9]. Moreover, they suffer from the same non-fulfillment as Handy1, since they cannot adapt if the user changes its pose during the meal or if the user is incapable of controlling the robot.

A recent review on robot feeding assistants research is described on [15]. In [7] it is proposed a system that uses Electroencephalography (EEG) signals and computer vision to help a user in the task of drinking using a robot arm. This system uses EEG signals to retrieve the user commands and a Kinect camera to track the user's mouth.

Following the same approach of using vision and EEG, in [16] it is proposed a feeding robotic system that uses EEG and a 4Degrees of Freedom (DOF) robot arm with a RGB camera attached to the end-effector. The user can choose between 3 different types of food using EEG signals. The robot arm is aligned with the user's mouth using the camera and doing visual servoing based on mouth detection, which uses Haar cascade classifiers. The adoption of EEG signals as user input to control the robot can be very useful for users that cannot control a joystick/keyboard buttons, for example, users that cannot move their fingers. However, the training for using these signals is required and sometimes is difficult to learn how to use them. Furthermore, a device has to be attached to the user's head to measure these signals which can be uncomfortable.

General purpose mobile robots, like PR2 [17], were also used in the development of feeding platforms. In [18] is proposed a feeding platform using PR2. This robot was controlled by the user using a web-based Graphical User Interface (GUI). By using a GUI the user could control the start and stop command of feeding sub-tasks, like scooping or re-scooping or the command to start the movement from the plate to the mouth. A depth camera is used to track the user's head and have access to the 3D position of the mouth. The tracking is done using an ARtag attached to the user's head. Using the same robot, research in multimodal anomaly detection was also done in [19].

Close research to the one that we present in this work was done in [20]. The authors propose to use DO[21] to track the user's head. However, the results do not report the accuracy of the tracking system or the time performance and do not test the whole platform in a real feeding environment.

Food Manipulation for feeding tasks was also investigated in [22], [23] and [24]. In [22] it is proposed a method for adapting non-optimal trajectories shown by the user to the optimal ones. This work can be used, for example, on adapting trajectories for Obi arm. Herlant's thesis [24] was focused on the strategies of food acquisition from the plate and on bite time prediction using social cues from group meal situations.

Robot	Path Planer	User MotionTracker	User Input Control	Type of Robot
Handy1 [8]	Predefined	No	Button	Research
Ishii et al [9]	Predefined	No	Head-Laser	Research
Obi [10]	User-Calibrated	No	Button	Commercial
MySpoon [11]	Predefined	No	Button	Commercial
MealBuddy [12]	Predefined	No	Button	Commercial
iEAT [13]	Predefined	No	Button	Commercial
Bestic [14]	Predefined	No	Button	Commercial
iCRAFT [25]	Predefined	No	Eye-Tracking	Research
Perera et al [16]	Real-Time	Yes	EEG	Research
Park et al [18]	Real-Time	Yes	Touch-GUI	Research
Herlant [24]	Predefined	Yes	Social Cues	Research

TABLE I: Feeding Robots Summary

We present in table I a summary of each one of the analyzed feeding robots. Each robot is classified in terms of the path planner, user motion tracker, user input control and the type of robot. The path planner can be: predefined, if the robot can only move to a predefined mouth position; user-calibrated, if the robot gives possibility to the user calibrate the mouth's position; or real-time adjusted, if the robot has some way of planing the motion from the plate to the mouth's position in real-time during the feeding. We also classify if the robot has some vision system for tracking the user's motion and what type of user input control it uses to give instructions to the robot. Finally, we classify if the robot is a commercial device or a research device.

B. Computer Vision Techniques for Feeding Robots

In our work, we are interested in building an autonomous robot capable of feeding a person. We are interested in being able to know the head pose of the person that is being fed and the mouth's location. In this section, we will make a brief review of the computer vision methods applied to this problem.

One way of localizing the face of the user is using a RGB image and a face detector to localize the face in the image. Several methods have been proposed in this area of face detection. One of the first works used a neural network to compute the bounding box of the faces present in the image [26]. Following this work, Viola and Jones [27] proposed a cascade classifier to classify different patches in the image. Viola and Jones face detector is still one of the most used face detectors due to the good performance in terms of speed and accuracy.

More recently, deep learning face detectors are the state of the art in face detection on RGB image. Works like MTCNN [28] or Face R-CNN[29] make use of Convolutional Neural Network (CNN)'s to achieve great accuracy and real-time performance. However, this performance comes with a high computational cost and, sometimes, a Graphics Processing Unit (GPU) is needed.

Another problem widely studied by the Computer Vision community, is the facial landmark detection and alignment problem [30] [31] [32] [33] [34] [35]. This problem is related to the alignment of pre-defined landmark points to the face that is present on the RGB image. Fig.1 illustrates the concept.

Recently, [30][31](OpenFace) describes a face analysis toolbox that goes beyond facial landmark detection, since it provides tools for head pose estimation, expression analysis,

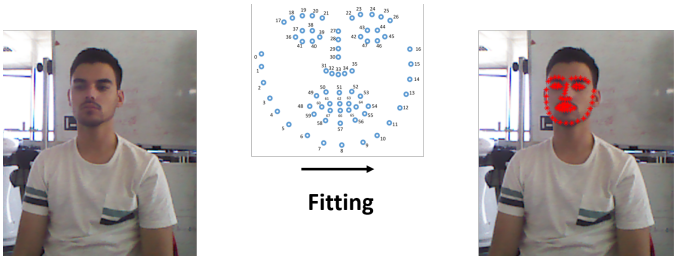


Fig. 1: Example of facial landmark alignment. In the facial landmark alignment problem we are interested in fitting a landmark model(present in the middle) to the face present on the image. We show, marked on red in the right image, the result of the alignment of the landmark model, using OpenFace[30].

and gaze estimation. All these tools are based on the facial landmark detection framework provided in [36]. OpenFace is of great value in facial expression analysis and it can be used in a wider field of applications like face analysis in healthcare systems or human-machine interaction interfaces. In our application, it can also be used to estimate the head pose of the user.

Cascade regression methods [37] have been applied to the task of face landmark detection and alignment [33] [32] showing great performance. These methods try to use a sequence of regressions to update the parameters of the model directly from features present in the image, for example, discovering the 2D pose (rotation and translation) of an object present in the image purely based in a sequence of regressors that accept as input features from the image. The regressors used can be, for example, a simple linear map between the features and an update in the pose parameters.

Supervised descent method [33] is an example of a cascade regression for face alignment, the authors motivate the regression as a supervised way of getting descent directions for solving non-linear least squares problems. In this work, the authors used Scale-Invariant Feature Transform (SIFT) features [38] and a linear regressor to update the landmark point positions. Using the same approach, Asthana et al.[34] propose to update the linear regressors in an online manner. The online update of the linear regressors provides a better performance in tracking scenarios since the updated regressors have the ability to adapt to the subject that is being tracked. Supervised descent method was also applied to dense 3D face alignment from 2D videos in [32], providing real-time performance in this task.

More recently, instead of using linear regressors, cascade CNN's were used [35][39] providing great performance, even in large and difficult poses. However, using CNN's come with the cost of high computational power requirements and worst performance in terms of time consumption.

Methods based on RGB data, e.g facial landmark alignment, have known issues when dealing with large poses. This happens since in large poses it is very difficult to detect the face on the RGB image, thus causing to be very difficult to estimate the head pose using this kind of approach. The solution to solve this problem can be the use of richer data like RGB-D data. However, there is a lack of research on this field and methods for head pose tracking rely on 3D model registration using, for example, Iterative Closest Point (ICP)[40] or its variants [41]. An exception is a method proposed by Fanelli et al. [42][43][44], this method uses depth data combined with regression forests to directly estimate the head pose with good accuracy.

Our work tries to incorporate the use of cascade regression methods, that show great performance on the RGB face alignment problem, in the problem of 3D head pose estimation. We formulate the problem of head pose estimation as a 3D rigid model registration and make use of DO[21][45][46], that show great results in the task rigid model registration, to the head pose estimation problem. In particular, we applied this work to the tracking of the head pose. DO, when compared to methods like ICP, shows better robustness to outliers, a larger region of convergence and real-time performance in the task of 3D rigid model registration[21].

III. Feedbot Architecture

In this work, we are interested in building an autonomous feeding robot arm system (Feedbot) with the purpose of feeding people with upper arm disabilities. We discussed that there are many commercially available robotic feeding platforms. However, none of those is capable of adapting, in real-time, to the changes of the pose of the users or to have feedback of the environment that surrounds the robot.

To give autonomy to Feedbot we rely on computer vision. We are interested in solving two problems. First, we want to be capable of having a visual feedback of the user's pose, more specifically the location of the mouth. Second, we want to have visual feedback on the spoon.

Having a visual feedback of the user's pose is important since people that live with disabilities can be incapable of moving their heads to a preset location, and using this visual feedback we can move the arm's end-effector to the correct mouth's position.

Visual feedback on the spoon is also important because it will allow to estimate the amount of food that is present on the spoon. Using this information we can, for example, detect if no food was scooped and re-scoop, or see if the user has already ate the food from the spoon.

To solve these two problems we propose a hardware setup for Feedbot that consists in 3 parts: a robot arm, a RGB camera pointing to a spoon attached to the robot arm end-effector and a RGB-D camera which is pointing to the user.

In fig. 2, we show our system successfully delivering food to a user, as well as the hardware setup that we used in our experiments.

Feedbot is capable of integrating the visual information, given by the RGB and RGB-D cameras, with the capability of movement provided by a robot arm. We divided our software into two parts, the vision system and the robot control system as it is illustrated in fig.3.

The vision system is responsible for two main tasks: tracking the user's mouth and provide feedback for what is present

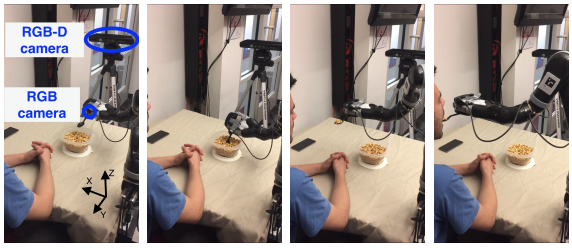


Fig. 2: Our feeding system successfully delivering food to a user. The first image also labels the axis orientations of the robot coordinate system.

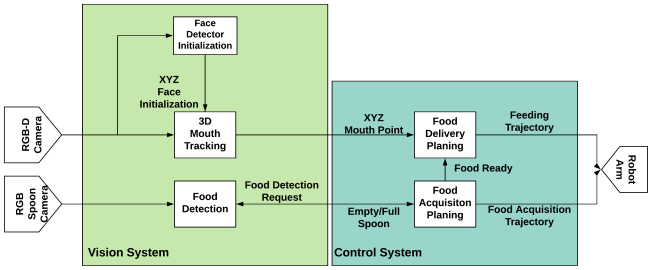


Fig. 3: System software architecture.

on the spoon. In the next section, we will introduce all the methods used to empower our system with these capabilities.

The control system is responsible for moving the robot arm to the location provided by the tracking and for acquiring food from the plate.

The control system of Feedbot is not the focus of this work. To give to Feedbot the capability of real-time adapting the position of the end-effector to the output provided by the vision system we use the approach described in [47]. Another task that the control system is responsible is the acquisition of food from the plate, to acquire food we use a recorded trajectory from the dataset [23], as described in [47].

IV. Feedbot's Perception

A. Head Pose Estimation

In order to work, Feedbot needs to know where the mouth is located. We are, then, interested in the problem of tracking the person's face. Tracking a person's face is a challenging task because faces are non-rigid objects and, in general, there is no good model to describe non-rigid objects. However, face movements are constrained by the head movements and the person's head is mostly a rigid body, so we will assume that the mouth is just a point in the rigid body shaped by the head. Using this assumption, our problem boils down to the head-pose estimation problem.

Given 2 point clouds $P_S \in \mathbb{R}^{3\times N_S}$ and $P_M \times \mathbb{R}^{3\times N_M}$, where P_M is a front view 3D model of the person's head and P_S is a 3D point cloud of the scene given by the RGB-D sensor. The 3D registration problem is known as fitting a rotation matrix R and a translation t that aligns the point cloud P_M with its correspondences in P_S. The fitting can be formalized by the following least-squares problem:

$$\begin{aligned} \underset{R,t,C}{\text{minimize}} \quad & \sum_{i=1}^{N_S}\sum_{j=1}^{N_M} C_{ij} \left\| p_S^i - R\cdot p_M^j - t \right\|^2 \\ \text{subject to} \quad & det(R) = 1; \quad R^T R = I; \\ & \sum_{i=1}^{N_S}\sum_{j=1}^{N_M} C_{ij} \geq N; \quad \forall j \sum_{i=1}^{N_S} Cij \leq 1; \quad \forall i \sum_{j=1}^{N_M} Cij \leq 1; \\ & C_{ij} \in \{0,1\}, \end{aligned} \tag{1}$$

where C is a matrix where each entry C_{ij} represents if the points p_S^i and p_M^j are correspondences. p_S^i is the point i of the point cloud P_S, and p_M^j is the point j of the point cloud P_M. Finally, N is the minimum number of correspondences that we want between the two point clouds.

When the correspondences between the model and scene point clouds are not known, it is a common practice to solve 3D registration problems using ICP [40]. Although ICP is a standard procedure to use, it tends to get trapped in local minima near the initialization provided and to have problems when dealing with a high percentage of outliers.

To overcome these problems we use Discriminative Optimization (DO) [21] to solve the model based 3D registration tracking problem presented in Equation 1. In [21] it is shown that DO is better than ICP in terms of computation time and more robust to outliers.

Discriminative Optimization (DO): DO [45] is a general framework to solve optimization problems when there is training data available. It can be seen as an iterative method that learns descent directions directly from training data. DO is a fast method compared to other approaches like ICP and has a wider convergence region. Moreover, since we want to deal with occlusions, DO is a good approach because it can handle a high percentage of outliers. Finally, DO has lighter computational requirements than approaches based on deep learning and can be run directly on the CPU.

DO is a learning-based methodology to tackle problems that are formulated as optimization problems but for which the function is unknown. From training data, DO learns a vector field, represented by a series of linear maps, that "emulate" the gradient of a function. During testing, given unseen data, DO follows this "gradient" leading to a stationary point that solves the optimization problem. Specifically, in our case, the algorithm is given a 3D model of the user's face that we want to track and generates a set of synthetic training data by applying random rotations and translations to the 3D model. This augmented training data is used to discover a sequence of linear maps that represent the descending directions. Those maps are used during inference, which is an iterative procedure that repeatedly updates the translation and rotation of the 3D model of the head to better align it with the the point cloud of the scene.

Initialization: 3D registration algorithms, like ICP and DO, require an initialization step that places the model of the face close to the face in the scene. We use a face detector to estimate an initial guess of where the face is in the RGB-D image. The difference between ICP and DO is that this

initialization can be worst when using DO since it has a much larger convergence region.

A popular face detector is the Viola Jones face detector [27]. Though this face detector is fast, it is not invariant to different face poses. Multi Task Cascaded Convolutional Neural Network (MTCNN) [28] is more robust to different face poses and also provides landmarks of the mouth, nose, and eyes. Despite MTCNN is based on deep learning, it can still run on a CPU at a lower speed. Since initialization does not happen frequently, the speed of the initialization step is not an important concern, and we remove the need for a GPU in our system by running initialization on a CPU.

We initialize our face tracker using the average 3D location of the facial landmarks given by MTCNN. The 3D locations of these landmark points can be computed because we are using an RGB-D camera. With these 3D points, we approximate the initial translation t^0 of the face model in the scene as:

$$\tilde{t} = \frac{1}{L}\sum_{i=1}^{L} p_S^i - \frac{1}{N}\sum_{j=1}^{N} p_M^j, \tag{2}$$

where L is the number of landmarks and N_M is the number of points in the model. For the initial rotation of the model, we assume $R^0 = I$, where I is the 3×3 identity. MTCNN landmarks can also be used to compute the mouth's 3D location in the model's point cloud which will be used as the 3D point to where the robot arm must move.

Acquiring a 3D face model: Our system must be capable of gathering, in an automatic way, the user's 3D face model. This is important because we want our system to deal with different users. To gather this 3D face model we used the approach described in fig.4.

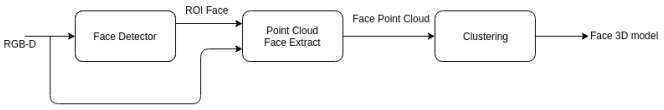


Fig. 4: Extraction of the 3D face model.

Given an image from the RGB-D sensor, we use a face detector such as Viola-Jones [27] or MTCNN [28]. The face detector provides a Region of Interest (ROI) containing the face of the user. We select, from the point cloud provided by the RGB-D sensor, the 3D points corresponding to the ROI Using the ROI provided by the face detector, sometimes, we get points that do not correspond to the user's face. These points come mainly from the background. To remove these points we used a clustering algorithm [48] and select the points corresponding to the cluster with the highest number of points, which provides the final 3D face model.

B. Camera-Arm Calibration

In order to the robot use the information from the RGB-D camera in a meaningful way, we have to calibrate the robot base frame with the RGB-D camera frame. We want to do this calibration in an automatic way, so that we can use Feedbot quickly after changing the position of the camera or the robot arm. We want to compute the rigid transformation (R, t) between the camera reference frame and the robot reference frame.

To automatically calibrate the robot frame with the camera frame, we attach an orange ball to the robot arm end-effector and then we move the robot arm to different positions. These positions are known in the robot base frame because we know the robot kinematics. For each of the positions, we acquire the corresponding RGB-D images from the camera. We use color segmentation to detect the ball in those images and get its corresponding 3D points. We fit a sphere to the 3D points of the ball using RANSAC to compute the 3D center of the ball in the camera frame. Finally, using these 3D correspondences between the camera frame and the robot base frame, we fit a rotation and a translation by solving an Orthogonal Procrustes Problem [49].

C. Food Detection

To see what is on the spoon, we place a tiny RGB camera on the end-effector of the robot arm. We use the images provided by this camera for two purposes, to detect if there is enough food to serve to the user and to detect if the user ate the food from the spoon. To detect if there is enough food on the spoon we use a detection algorithm that we can tune to specify the required amount of food, to do this we need to have a correspondence between the food image and the weight of food. To detect if the user has eaten the food, we define a classifier with two classes: "empty" and "non-empty" spoon, which receives as input the images from the spoon's camera.

The first problem to solve is that the image provided by the camera does not contain only the spoon. To solve this problem, we developed a calibration step that computes a mask of the spoon's location. Calibration is performed by acquiring an image with a white sheet of paper under the spoon. Otsu's method [50] is used to select a threshold and segment the image which provides a spoon's mask. This method works well because the colored spoon contrasts sharply with the white sheet of paper. Using this calibration step, our camera will provide images like the one present in fig. 5.

Fig. 5: Example masked images of the plastic spoon, showing the spoon empty, with nuts, and with rice.

Food Weight Estimation: We need to have a measure of how much food is present on the spoon in order to be capable of deciding when is enough food to deliver to the user or not.

In the calibration step, we also record a histogram, H_s, of the H channel of the HSV image of the empty spoon after applying the calibration mask. The recorded histogram, H_s, is compared with the histograms of subsequent images of the spoon, H_a. To compare the histograms, we used the normalized correlation between the two histograms,

$$d(H_s, H_a) = \frac{\sum_{i=1}^{N}(H_s(i) - \bar{H_s})(H_a(i) - \bar{H_a})}{\sqrt{\sum_{i=1}^{N}(H_s(i) - \bar{H_s})^2 \sum_{i=1}^{N}(H_a(i) - \bar{H_a})^2}}, \quad (3)$$

where $\bar{H} = \frac{1}{N}\sum_{i=1}^{N} H(i)$ is the histogram mean. If the value of the correlation is below a certain threshold, then the algorithm reports that there is enough food on the spoon.

We train a linear regression to discover the relationship between the histogram correlation value and the weight of the food on the spoon. This regression was trained using a dataset containing examples of spoons with food and the correspondent weight. This regression can be used to inform the choice of the threshold for "enough food" on the spoon.

Empty/Non Empty Spoon Classification: Another task that Feedbot must be capable of is to distinguish between an empty spoon and a non-empty spoon. This is important since we do not want to present empty spoons to the user. It is also important because we can know when the user has finished eating the food present on the spoon.

We want to classify the masked images gathered by the camera in two classes, to classify these images we tried two different classifiers: a linear Support Vector Machine (SVM) and a logistic regression.

The input feature to the logistic regression is the value of the histogram correlation between the current spoon and an example of an empty spoon, and the input features to the SVM are the bins of the H color channel histogram of the spoon image.

V. Feedbot's Experimental Evaluation

A. Feedbot Evaluation

We tested Feedbot using two different robot arms, namely Kinova MICO and Niryo One[2], showing that Feedbot system can use different robot arms. Kinova MICO is a 6DOF commercially available robot arm that is highly used by wheelchairs users. Niryo One is a 6DOF low cost and easy to use 3D printed robot arm, the whole system is open source and fully compatible with Robot Operating System (ROS). It is commercially available, but can also be constructed since the hardware is open source.

Kinova MICO Setup: To test our Feedbot and validate the whole system components in a real meal scenario, we did a small experiment with two different persons [1]. In this experiment, we used as robot arm a Kinova MICO, the setup used in the experiment can be seen in fig.2.

In this study, we used three different types of food, namely, peanuts, M&M's and mac'n cheese. We used different types of food in order to test our system in different aspects. Peanuts were used since they are difficult to feed without falling off the spoon if the robot end-effector is too shaky, in fig.6 we present an example of Feedbot feeding peanuts to one of the persons in the experiment.

To test the capability of Feedbot to deal with foods of different colors we used M&M's. M&M's are good to test this aspect since each of them has a different color and together they form a very unusual color pattern, this pattern was good to test if the RGB camera that looks to the spoon, and detects if there is enough food on the spoon, can still work with different color patterns. In fig.7, we present an example of Feedbot feeding M&M's to the second subject.

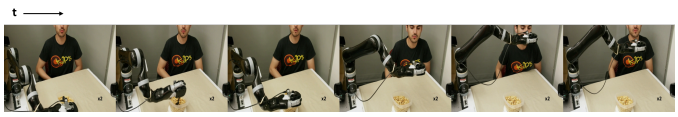


Fig. 6: Feedbot feeding peanuts.

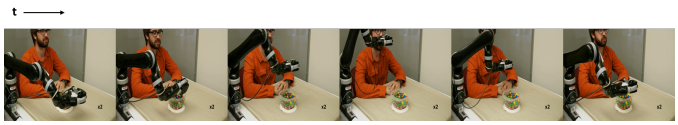


Fig. 7: Feedbot feeding M&M's.

We confirmed that our procedure for getting the feedback of how much food is present on the spoon has some drawbacks since it is based on the comparison of the color histogram. In cases where we got many M&M's with the same color as the spoon, we got some false empty spoons detections.

Using mac'n cheese as testing food, we were capable of test the Feedbot's acquisition feedback, since mac'n cheese is difficult to scoop and sometimes gets stick to the plate. As we can see in fig.8 when the scoop movement fails to acquire the food from the plate, our feedback system is capable of reporting that the spoon is empty and Feedbot tries to scoop again.

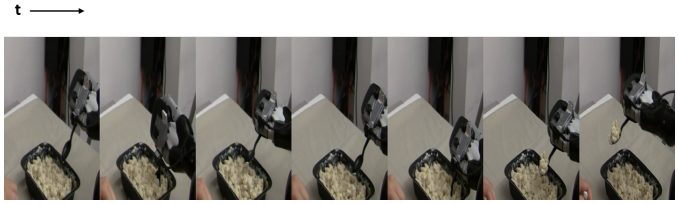


Fig. 8: Feedbot feeding mac'n cheese.

Niryo One Setup: To validate that Feedbot system works with different robot arms, we did an experiment using a Nyrio One robot arm [2]. In this experiment, we tested Feedbot with one user that suffer from upper-extremity disabilities, in this test we were more focused on testing the capability of Feedbot to track and adapt its trajectories to a user that is moving in real-time.

In fig.9, it is shown that Feedbot is capable of adapting its trajectory to the user that is moving.

The aforementioned tests were not thought to exhaustively test Feedbot, the purpose of these tests was to do a preliminary test to the practicability of Feedbot in a real meal scenario, more tests to the robustness of Feedbot will be done in future

[1] In https://www.youtube.com/watch?v=X7McqWk1AK8 it is available a video showing Feedbot performing different tasks on this experiment

[2] In https://bit.ly/2Evpkf1 it is available a video showing Feedbot performing real-time trajectory adaptation in this experiment

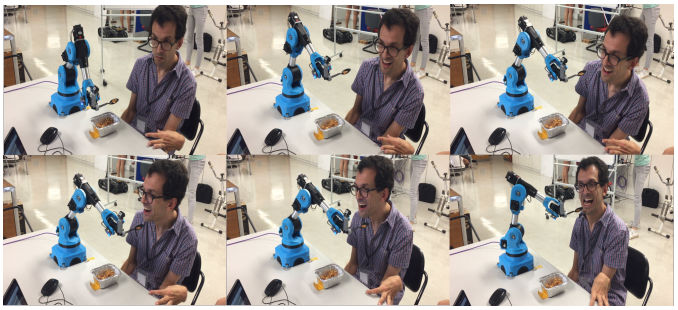


Fig. 9: Feedbot feeding moving subject.

work. However, these preliminary tests confirm that Feedbot is capable of handle simple food types and can be used by different persons. Furthermore, it is also confirmed that the different parts of Feedbot can work together and that Feedbot works with different robot arms.

B. 3D Head Pose Tracking

Offline Validation: To validate the performance of Feedbot's tracking system, we used two public available RGB-D datasets for head pose estimation, namely, Biwi[42] and ICT3DHP [51]. The evaluation on these datasets was done offline, this means that we give to our algorithm as much time as it needs to process each frame and that no frame is missed.

We tested our algorithm against two state of the art methods for 3D head-pose estimation, namely, OpenFace [30] and Fanelli et al. method [42]. In our tests, we used the implementation provided by the authors with the default parameters.

In all our experiments, we acquired a 3D head model using the method described in section IV-A. This model was acquired using the first frame on each of the dataset sequences.

Biwi dataset is composed by 24 RGB-D sequences of 20 different people (4 people were recorded twice). Biwi contains a wide range of head poses and the ground truth is given in the form of the 3D head position and rotation in the camera frame. This dataset was acquired with frame by frame estimation in mind and not for tracking, which means that some frames are missing. However, this dataset is widely used on head pose tracking benchmarks, mainly to test algorithms in difficult scenes [31], since there is a higher than normal temporal gap between some frames.

Due to the difficult nature of this dataset, DO fails to converge in 4 of the 24 sequences. We removed these sequences from the dataset in the comparison against OpenFace and Fanelli et al. OpenFace[30] has a failure detection step, thus it can recover from failure cases which is not possible with DO since we do not have a failure detection step. Fanelli's method[42] is based on frame by frame estimation, therefore, when it fails it simply starts again in the next frame and does not accumulate the error as DO.

The results for the 20 valid sequences, in terms of the MAE for the Euler angles of the head rotation are presented in table II and for the head position in table III.

We can conclude that our method outperforms state of the art methods in terms of the MAE for the head rotation Euler angles. The analysis of the results show that our method has higher MAE for the Ry Euler angle. It was expected since is in this direction that people have a wider amplitude of movements with their heads. Moreover, we conclude that OpenFace has more than twice the error that our method has in the Rx direction, this can be explained due to the occlusion of the face when people rotate their heads in this direction. For the head position, we conclude that our method can still be competitive relative to other state of the art methods and provides a good accuracy with a MAE less than 1 cm.

Method	Rx	Ry	Rz	Mean
DO(ours)	**2.83**	6.60	**4.70**	**4.71**
OpenFace [30]	7.55	6.12	10.02	7.90
Fanelli et al. [42]	3.91	**4.88**	10.69	6.49

TABLE II: MAE Euler angles in degrees for the Biwi dataset.

Method	Tx	Ty	Tz	Mean
DO(ours)	0.84	0.75	1.03	0.87
OpenFace [30]	1.07	0.96	3.21	1.74
Fanelli et al. [42]	**0.32**	**0.28**	**0.27**	**0.29**

TABLE III: MAE head position in cm for the Biwi dataset.

ICT3DHP[51] is composed by 10 RGB-D sequences of 10 different people. It was acquired with the purpose of benchmarking head pose tracking methods, however, the ground truth is only precise for the rotation of the head. The head annotated position is referenced by the authors as not precise, due to this we only present results for the estimation of the head rotation.

We present in table IV the MAE for the rotation Euler angles in the ICT3DHP dataset.

Method	Rx	Ry	Rz	Mean
DO(ours)	**3.06**	6.88	2.89	4.28
OpenFace [30]	3.23	**3.44**	**2.88**	**3.19**
Fanelli et al. [42]	7.76	8.27	7.69	7.91

TABLE IV: MAE Euler angles in degrees for the ICT3DHP dataset.

Our results in ICT3DHP dataset confirm the consistency and competitive performance of our method across different datasets. We can conclude that the performance of Fanelli et al. is worst than in the Biwi dataset, this can be explained due to the use of the default parameters set by the authors, these parameters were tuned for the Biwi dataset. Although OpenFace has the better MAE results in this dataset, we can see that in the Rx direction DO still has the better results, thus confirming our, previously discussed, idea that OpenFace fails when dealing with face occlusions.

The evaluation of DO across these two different datasets reveals that it is capable of achieving better, or equal performance when compared with other state of the art head pose estimation algorithms. Moreover, DO reveals a better generalization to different datasets, is not sensitive to parameter tuning and is capable of dealing with extreme face occlusions.

Real-Time Validation: To test the performance of our tracking system in real-time, we acquired a RGB-D video of a person moving his head in front of the camera. Using ROS, we ensure that we can simulate the real-time application. Using

the recorded video if the tracking takes more computation time than the camera acquisition time, some frames are skipped.

The error of our tracking system was measured, in terms of absolute error and MAE, by annotating the mouth point in our dataset at a frame-rate of 2 FPS. The error in each frame is presented in fig.10. The experiment was performed on a single thread on a Intel Core i7-6500U CPU 2.50GHz with 8GB of memory. Our tracking achieved speeds of 28FPS.

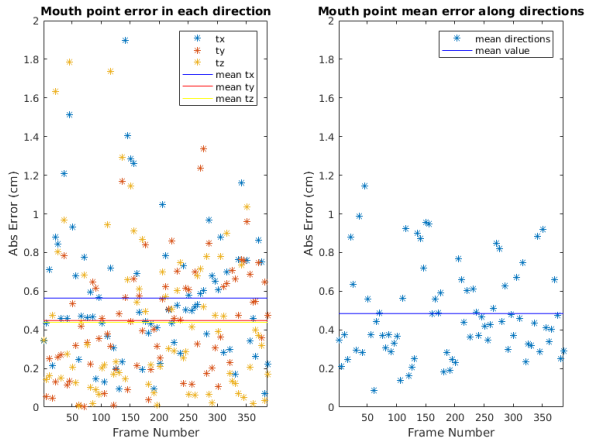


Fig. 10: Tracking error of DO in each frame. On the left, the asterisks represent the absolute value of the error in each frame for each of the directions, the lines represent the MAE along the sequence for each of the directions. On the right, the asterisks represent the MAE for each of the frames and the line represent the mean of these values along all frames of the sequence.

Fig.10 shows that our tracking method can track the person's mouth with an absolute error below 2cm through the entire video. This is on the order of magnitude of the mouth's size, so the output of our system will not be far from the real location of the mouth. These results are of the same order of magnitude as the ones present on table III, thus demonstrating the consistency of our results.

To have a better visual insight about the output of our tracking method, we show in red in fig.11 the output mouth point of our algorithm.

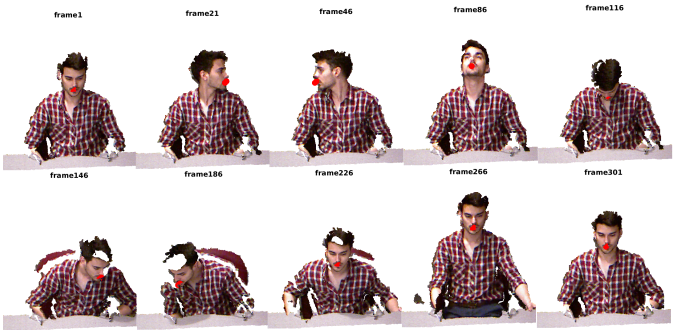


Fig. 11: Tracking output for the user's mouth.

C. Food Detection

In this section, we evaluate the performance of our two food detection methods. To train the classifiers and the linear regression we used scikit-learn[52]. We split our datasets into 70% train data and 30% test data. To evaluate and train our algorithms for food detection, we gathered two small datasets, one containing 387 images labeled with the weight of food and another one with 119 images of empty spoons and 118 images of non-empty spoons.

Food Weight Estimation: Fig.12 shows the the linear regression fit modeling the weight of food in the spoon with the correlation of histograms between the current image of the spoon and the image of the empty spoon. The correlation measure is computed, as shown in Equation 3, by taking the histogram of the image of the spoon with food on it and correlating that with the histogram of the baseline image of the spoon without food on it.

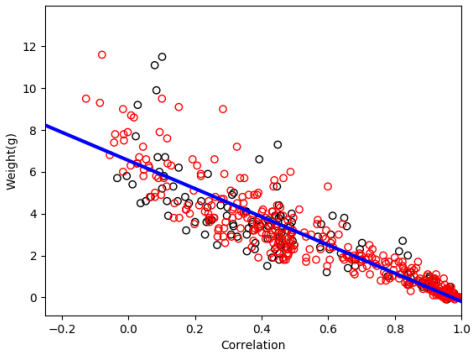


Fig. 12: Relationship between the correlation measure and the weight of food on the spoon. The data points used for training are shown in red, the data used for test is shown in black, and the linear regression is shown in blue.

The regression has a mean squared error of 1.37g and a correlation coefficient (r^2) of 0.75 on the test set. There is a strong negative correlation between the weight of the food that is on the spoon and the correlation between the histograms. Even if the model is not perfect in measuring the amount of food that is on the spoon, we can use this model to calibrate our correlation threshold to require more or less food per bite.

Empty and Non-empty Spoon Classification: In the classification task, we achieved an accuracy of 95.8% with the logistic regression and 98.6% with the linear SVM. The SVM was trained using the whole H channel histogram and the logistic regression using only the correlation between each sample of the train data and an empty spoon histogram example. We present, in fig.13 and fig.14, the confusion matrix results for the two classification methods.

These results validate our classification approach to discover if there is still food on the spoon. More interestingly, looking at the confusion matrix for the linear SVM (fig.14), we can conclude that it will not classify an empty spoon as a full spoon. This is very important since it means that using this approach Feedbot will not give empty spoons to the user.

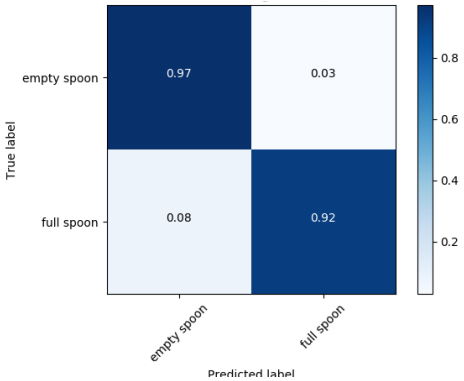


Fig. 13: Confusion Matrix for Logistic Regression

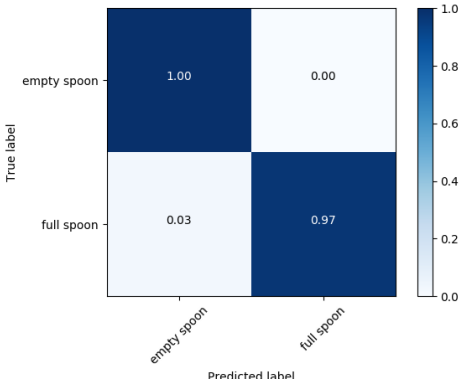


Fig. 14: Confusion Matrix for SVM

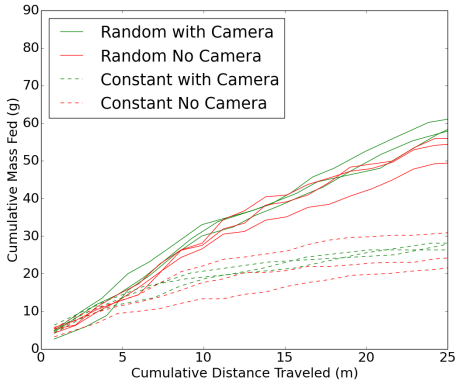
D. Ablation Study Food Acquisition

Setup: To analyze the importance of various components of Feedbot, we perform an ablation study in the feeding task where we measure the efficiency of Feedbot when it does and does not use certain food acquisition strategies. We define the efficiency to be the mass of food that is delivered to the feeding location as a function of the total distance that the tip of the spoon has traveled. We report the mass as a function of distance traveled rather than the time taken because we want to negate any effect that setting the robot to a faster or slower speed would have on the results.

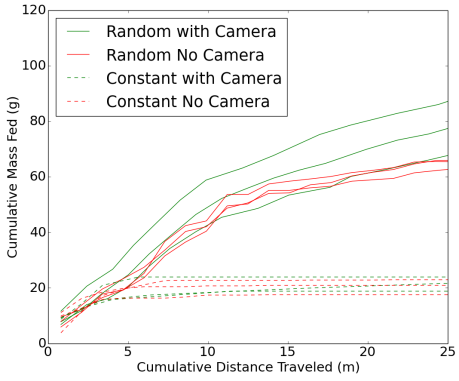
The Feedbot components that we alter in our ablation study are 1) whether or not we re-scoop if the spoon-facing camera detects that not enough food was acquired and 2) whether Feedbot always scoops from the same position on the serving plate or whether it scoops from a uniformly random position within a 6cm x 3cm rectangle on the plate. We perform the ablation experiment on two different kinds of food: peanuts and fried rice. In this way, we can determine if the importance of system components depends on the type of food.

For consistency in results, we use the same random seed for all trials that randomize the scooping location on the plate. To speed up data collection, the robot dumps the food directly onto the scale after food acquisition. Then, in our data analysis, we add in the distance to the user's mouth and back. Cutting out the travel time between the plate and the user's mouth cuts the data acquisition time by a factor of three On average, the distance traveled by the spoon tip in a single scooping motion is 32cm and the average distance to and from the user's mouth is 49cm in each direction. We use these average distance values when representing the amount of food served as a function of distance.

Results: In our ablation study for food acquisition, we find that vision feedback significantly increases the amount of food brought to the user in each bite. For the spoon-facing vision feedback system, we use a histogram correlation cutoff of 0.5 for both rice and nuts. For rice, when randomizing the scooping location, the average amount of rice served per bite in the first 20 bites across three trials is 4.8g with camera feedback and 3.2g without camera feedback. Likewise, for nuts it is 3.8g with camera feedback and 2.7g without. There is an added distance that the end-effector travels in re-scooping when using camera feedback. Despite this added cost in distance, fig.15 shows that even when looking at the amount of food served as a function of the total distance that the spoon tip travels (including the added distance required to re-scoop), after 25m random scooping with camera feedback consistently outperforms random scooping without camera feedback.



(a) Feeding Nuts



(b) Feeding Rice

Fig. 15: Charts showing the amount of food fed (in grams) as a function of the distance the tip of the spoon has traveled, including the distance traveled in re-scooping. The chart includes results for scooping location randomization (solid lines) compared to a constant scooping location (dotted lines), and it includes results with (green lines) and without (red lines) camera feedback on whether food was successfully acquired. We ran three trials for each test type.

For both rice and nuts we find that randomization of the scooping location is a useful technique in food acquisition. Fig.15 also shows that randomization coupled with visual feedback on the success of a scoop is the most effective. Most interestingly, we find that the importance of our system components depends on the type of food used. We find that randomizing the location of scoops is more important when serving rice instead of nuts. This could be related to the fact that nuts were observed to "settle" after each scoop, whereas fried rice will tend not to fill in the hole left after scooping. Thus, scooping the same place for fried rice will quickly result in very little (almost no) mass acquired by the spoon in subsequent scoops.

VI. Conclusions and Future Work

In this work, we explored how vision can be introduced in a robotic feeding platform. By introducing a vision-based approach in Feedbot, we made it more aware of the environment and more autonomous, making it capable of adapting its trajectories to the user in real-time. Feedbot is fully capable of feeding different types of food to a user using a spoon. We tested Feedbot using two different robot arms, Kinova MICO and Niryo One, proving that it can be used in different types of robot arms. Using a robot arm like Niryo can help us to highly deploy our Feedbot in institutions, due to the low cost of this arm.

We applied DO to the task of head pose tracking showing that this method is capable of achieving the same, or even better, performance than state of the art methods. DO was evaluated in terms of the speed performance achieving real-time performance, thus showing its potential to be used for real-time applications like robotics. However, in our tests, we also understood the limitations of using DO for tracking without using a validation step. Future work will be done to improve the robustness of our head pose tracking method.

Food detection methods, based on visual information, were introduced in this work. We found that using very simple models, based on visual information, can achieve good performance. This is very important, because these methods, due to their simple nature, can be applied in real-time since they do not have high computational cost.

Using an ablation study, we measured the impact of visual feedback in the food acquisition step. We show that the use of visual feedback significantly impacts the performance of robotic feeding platforms in terms of the food delivered to the user. This is an important finding, on one hand, it shows the need to have food detection methods in feeding robots, on the other hand, it shows that current commercial available feeding robots are not efficient since they do not have any type of feedback in the food acquisition step.

Future work in modeling the user's intention needs to be done. We remember that some of the transitions, that are related to the detection of the user's intention, are still time-based. Modeling the user's intention is a very difficult problem since, to the best of our knowledge, there are no good mathematical models for that. A possible way of contributing for solving this problem could be based on the user facial expression, pose, and some type of supervised signals from caregivers (e.g trajectories of the spoon during a meal), try to detect patterns that reveal the user's intention to be feed. Another topic that still needs research is the problem of controlling and adapting the trajectory of the robot as the plate gets empty. This can be done, for example, using visual feedback on the plate and on the spoon.

Finally, we are planning to test and evaluate Feedbot in meal scenarios with more people that suffer from upper-extremity disabilities. We plan to to quantitatively and qualitatively analyze the efficacy of Feedbot. We will also compare Feedbot to the type of baseline in current commercially-available robots.

References

[1] R. K. Al-Halimi and M. Moussa, "Performing Complex Tasks by Users with Upper-Extremity Disabilities Using a 6-DOF Robotic Arm: A Study," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 6, pp. 686–693, 6 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7553526/

[2] "Niryo Smart Collaborative Robot For You." [Online]. Available: https://niryo.com/

[3] D. Feil-Seifer and M. J. Matarić, "Defining socially assistive robotics," in *Proceedings of the 2005 IEEE 9th International Conference on Rehabilitation Robotics*, vol. 2005. IEEE, 2005, pp. 465–468. [Online]. Available: http://ieeexplore.ieee.org/document/1501143/

[4] S. D. Klee, B. Q. Ferreira, R. Silva, J. P. Costeira, F. S. Melo, and M. Veloso, "Personalized assistance for dressing users," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9388 LNCS, pp. 359–369, 2015.

[5] Z. Erickson, M. Collier, A. Kapusta, and C. C. Kemp, "Tracking Human Pose During Robot-Assisted Dressing using Single-Axis Capacitive Proximity Sensing," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2245–2252, 9 2017. [Online]. Available: http://arxiv.org/abs/1709.07957

[6] K. P. Hawkins, P. M. Grice, T. L. Chen, C. H. King, and C. C. Kemp, "Assistive mobile manipulation for self-care tasks around the head," in *IEEE SSCI 2014 - 2014 IEEE Symposium Series on Computational Intelligence - CIR2AT 2014: 2014 IEEE Symposium on Computational Intelligence in Robotic Rehabilitation and Assistive Technologies, Proceedings*, 2014, pp. 16–25.

[7] S. Schröer, I. Killmann, B. Frank, M. Völker, L. Fiederer, T. Ball, and W. Burgard, "An autonomous robotic assistant for drinking," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June. IEEE, 5 2015, pp. 6482–6487. [Online]. Available: http://ieeexplore.ieee.org/document/7140110/

[8] M. Topping, "Early experience in the use of the Handy 1 robotic aid to eating," *Robotica*, vol. 11, no. 06, p. 525, 11 1993. [Online]. Available: http://www.journals.cambridge.org/abstract_S0263574700019366

[9] S. Ishii, S. Tanaka, and F. Hiramatsu, "Meal assistance robot for severely handicapped people," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 1995, pp. 1308–1313. [Online]. Available: http://ieeexplore.ieee.org/document/525461/

[10] "Obi — Robotic feeding device designed for home care." [Online]. Available: https://meetobi.com/

[11] "SECOM - Meal-assistance Robot." [Online]. Available: https://www.secom.co.jp/english/myspoon/

[12] "Meal Buddy," p. 3. [Online]. Available: https://www.performancehealth.com/meal-buddy-systems

[13] "Assistive Innovations - iEAT Feeding Robot." [Online]. Available: https://www.assistive-innovations.com/en/eatingdevices/ieat-feeding-robot

[14] "Comanio Care: Bestic." [Online]. Available: https://www.camanio.com/us/products/bestic/

[15] I. Naotunna, C. J. Perera, C. Sandaruwan, R. Gopura, and T. D. Lalitharatne, "Meal assistance robots: A review on current status, challenges and future directions," in *2015 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 12 2015, pp. 211–216. [Online]. Available: http://ieeexplore.ieee.org/document/7404980/

[16] C. J. Perera, T. D. Lalitharatne, and K. Kiguchi, "EEG-controlled meal assistance robot with camera-based automatic mouth position tracking and mouth open detection," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 5 2017, pp. 1760–1765. [Online]. Available: http://ieeexplore.ieee.org/document/7989208/

[17] "Overview — Willow Garage." [Online]. Available: http://www.willowgarage.com/pages/pr2/overview

[18] D. Park, Y. K. Kim, Z. M. Erickson, and C. C. Kemp, "Towards Assistive Feeding with a General-Purpose Mobile Manipulator," Tech. Rep., 5 2016. [Online]. Available: http://arxiv.org/abs/1605.07996

[19] D. Park, H. Kim, and C. C. Kemp, "Multimodal anomaly detection for assistive robots," *Autonomous Robots*, pp. 1–19, 2018. [Online]. Available: https://doi.org/10.1007/s10514-018-9733-6

[20] C. Silva, J. Vongkulbhisal, M. Marques, J. P. Costeira, and M. Veloso, "Feedbot - A robotic arm for autonomous assisted feeding," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10423 LNAI, pp. 486–497, 2017. [Online]. Available: http://link.springer.com/10.1007/978-3-319-65340-2_40

[21] J. Vongkulbhisal, F. De La Torre, and J. P. Costeira, "Discriminative optimization: Theory and applications to point cloud registration," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 3975–3983, 2017. [Online]. Available: http://arxiv.org/abs/1707.04318

[22] T. Rhodes and M. Veloso, "Robot-driven Trajectory Improvement for Feeding Tasks," in *IEEE Internation Conference on Intelligent Robots Robots*, 2018. [Online]. Available: http://www.cs.cmu.edu/~mmv/papers/18iros-RhodesVeloso.pdf

[23] T. Bhattacharjee, H. Song, G. Lee, and S. S. Srinivasa, "Food manipulation: A cadence of haptic signals," 2018. [Online]. Available: http://arxiv.org/abs/1804.08768

[24] L. V. Herlant, S. S. Srinivasa, C. G. Atkeson, C. Ri, J. Forlizzi, C. Hcii, L. A. Takayama, and S. Cruz, "Algorithms, Implementation, and Studies on Eating with a Shared Control Robot Arm," Ph.D. dissertation, 2018. [Online]. Available: http://www.cs.cmu.edu/afs/cs/user/lcv/www/herlant-thesis.pdf

[25] P. Lopes, R. Lavoie, R. Faldu, N. Aquino, J. Barron, M. Kante, B. Magfory, and W. Meleis, "Eye - Controlled Robotic Feeding Arm Technology (iCRAFT)," pp. 1–24, 2011. [Online]. Available: http://www.ece.neu.edu/personal/meleis/icraft.pdf

[26] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 1, pp. 23–38, 1998. [Online]. Available: http://www-prima.imag.fr/jlc/Courses/2016/PRML/rowley-ieee-PAMI.pdf

[27] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, pp. I–511–I–518, 2001. [Online]. Available: http://ieeexplore.ieee.org/document/990517/

[28] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 4 2016. [Online]. Available: http://arxiv.org/abs/1604.02878http://dx.doi.org/10.1109/LSP.2016.2603342

[29] H. Wang, Z. Li, X. Ji, and Y. Wang, "Face R-CNN," Tech. Rep., 2017. [Online]. Available: https://arxiv.org/pdf/1706.01061.pdfhttp://arxiv.org/abs/1706.01061

[30] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency, "OpenFace 2.0: Facial Behavior Analysis Toolkit," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, 5 2018, pp. 59–66. [Online]. Available: https://ieeexplore.ieee.org/document/8373812/

[31] T. Baltrusaitis, P. Robinson, and L. P. Morency, "OpenFace: An open source facial behavior analysis toolkit," in *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*. IEEE, 3 2016, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/document/7477553/

[32] L. A. Jeni, J. F. Cohn, and T. Kanade, "Dense 3D face alignment from 2D videos in real-time," in *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, FG 2015*, no. MAY, 2015. [Online]. Available: http://lightside.hu/pub/articles/Jeni15FG_ZFace.pdf

[33] X. Xiong and F. De La Torre, "Supervised descent method and its applications to face alignment," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 6 2013, pp. 532–539. [Online]. Available: http://ieeexplore.ieee.org/document/6618919/

[34] A. Asthana, S. Afeiriou, S. Cheng, and M. Pantic, "Incremental Face Alignment in the Wild," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2014. [Online]. Available: https://ibug.doc.ic.ac.uk/media/uploads/aasthanacvpr2014.pdf

[35] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, "Face Alignment Across Large Poses: A 3D Solution," in *CVPR IEEE Conference*, 2016. [Online]. Available: http://www.cbsr.ia.ac.cn/users/

[36] T. Baltrušaitis, P. Robinson, and L. P. Morency, "Constrained local neural fields for robust facial landmark detection in the wild," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 354–361. [Online]. Available: https://www.cl.cam.ac.uk/~tb346/pub/papers/iccv2013.pdf

[37] P. Dollár, P. Welinder, and P. Perona, "Cascaded pose regression," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1078–1085. [Online]. Available: https://pdollar.github.io/files/papers/DollarCVPR10pose.pdf

[38] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. [Online]. Available: https://link.springer.com/content/pdf/10.1023%2FB%3AVISI.0000029664.99615.94.pdf

[39] X. Zhu, X. Liu, Z. Lei, and S. Z. Li, "Face Alignment In Full Pose Range: A 3D Total Solution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/8122025/

[40] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[41] S. Gold, C. P. Lui, and A. Rangarajan, "New Algorithms for 2D and 3D Point Matching : Pose Estimation and Correspondence," *Pattern Recognition*, vol. 31, no. 8, 1999. [Online]. Available: https://papers.nips.cc/paper/977-new-algorithms-for-2d-and-3d-point-matching-pose-estimation-and-corresponden pdf

[42] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. van Gool, "Random Forests for Real Time 3D Face Analysis," *International Journal of Computer Vision (IJCV)*, vol. 101, no. 3, pp. 437–458, 2013. [Online]. Available: https://data.vision.ee.ethz.ch/cvl/gfanelli/pubs/ijcv.pdf

[43] G. Fanelli, T. Weise, J. Gall, and L. Van Gool, "Real Time Head Pose Estimation from Consumer Depth Cameras," in *Proceedings of the DAGM Symposium on Pattern Recognition*, 2011, pp. 101–110. [Online]. Available: https://data.vision.ee.ethz.ch/cvl/gfanelli/pubs/dagm11.pdfhttp://link.springer.com/10.1007/978-3-642-23123-0_11

[44] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *Computer Vision and Pattern Recognition*, 2011, pp. 617–624. [Online]. Available: https://data.vision.ee.ethz.ch/cvl/gfanelli/pubs/cvpr11.pdfhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5995458

[45] J. Vongkulbhisal, F. De La Torre, and J. P. Costeira, "Discriminative Optimization: Theory and Applications to Computer Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8338139/

[46] J. Vongkulbhisal, "Discriminative Optimization: Theory and Applications to Computer Vision Thesis," Ph.D. dissertation, CMU, 2018. [Online]. Available: http://users.isr.ist.utl.pt/~jpc/ftp/thesisjayakorn.pdf

[47] A. Candeias, T. Rhodes, M. Marques, J. P. Costeira, and M. Veloso, "Vision Augmented Robot Feeding," in *Sixth International Workshop on Assistive Computer Vision and Robotics in Conjuction with ECCV2018 (accepted)*, 2018, pp. 1–16.

[48] M. Daszykowski and B. Walczak, "Density-Based Clustering Methods," *Comprehensive Chemometrics*, vol. 2, pp. 635–654, 2010. [Online]. Available: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[49] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3-D rigid body transformations: a comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, pp. 272–290, 1997. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.173.2196&rep=rep1&type=pdf

[50] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979. [Online]. Available: http://ieeexplore.ieee.org/document/4310076/

[51] T. Baltrusaitis, P. Robinson, and L. P. Morency, "3D Constrained Local Model for rigid and non-rigid facial tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2610–2617. [Online]. Available: http://www.cl.cam.ac.uk/research/rainbow/emotions/

[52] P. A. C. D. B. M. P. M. D. E. Vanderplas, J., "Scikit-learn: Machine learning in Python." *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.