The Art and Craft of Making the Tortellino: Playing with a Digital Gesture Recognizer for Preparing Pasta Culinary Recipes

M. ROCCETTI, G. MARFIA
University of Bologna, Bologna, Italy
AND
M. ZANICHELLI
Onde Comunicazione, Bologna, Italy

The innovative aspects introduced by the new hands-free gaming systems, like the Nintendo Wii, Sony Move and Microsoft Kinect, indicate that technology is progressively, and at a limited cost, reaching more natural ways of interacting with human beings, and vice versa. This process can clearly pave the way to beneficial opportunities in a wealth of fields, including those of technical and artistic interactive exhibitions. In fact, while for long all types of tech-inspired exhibitions have been mainly based on artifacts and tools that seldom required any interactions with the public, today this can radically change. The new frontiers of technology for performing arts provide the means for creating active interactions between the public and any object, allowing visitors to enjoy an experience that exceeds what can be solely passively seen or heard. In such domain, we describe the design and implementation of a multimedia system that interactively illustrates the process of preparing one of the masterpieces of the culinary heritage of the Italian city of Bologna, the *Tortellino*. Step-by-step, with our system, anyone can enjoy a virtual experience learning how to prepare a *Tortellino* while mimicking the movements a real cook would perform to prepare its recipe when starting from its raw ingredients: eggs and flour. From a scientific viewpoint, the challenging part of this project resides in recognizing cooks' movements while verifying their correctness. In this paper, we describe how we achieved such result devising the most appropriate techniques able to allow our system to recognize a predefined set of actions (as those performed by a real cook) with the help of no other external hardware device, but a simple camera. Witnessing the importance of the results we got, our multimedia system has been chosen to be part of the expositions that will represent the City of Bologna at the Shanghai Universal Expo, in October 2010.

Categories and Subject Descriptors: J 5 **[Arts and Humanities**] – Arts, Fine and Performing.

General Terms: Design, Performance

Additional Key Words and Phrases: Multimedia and Interactive Systems, Performing Arts, Technology-based Exhibitions, Gesture Following, Hands-Free Technology, Italian Pasta, Tortellini

1. INTRODUCTION

The *Tortellino* is beyond any doubt a symbol of Bologna and of the Italian cuisine, and as such it can be generally found on the menus of the most renowned Italian restaurants around the world. Essentially, for those who do not know it, the *Tortellino* is a dumpling usually stuffed with meat

Contact Author's address: M. Roccetti, Department of Computer Science, University of Bologna, Italy. E-mail: roccetti@cs.unibo.it.

and cheese. In spite of its fame, only few know about its long and fascinating story. An old legend says that the *Tortellino* was modeled on the belly button of Goddess Venus. Historical studies, instead, trace back the origin of the *Tortellino* to a tradition which became popular in the Middle Ages in the area of Bologna, eating hand rolled egg pasta and meat broth; a common usage that derived from the presence of a successful cattle, poultry and capon raising industry [Capatti *et al*. 2003].

Although it is possible to find many references, especially during the Renaissance period, to specialties that may be recognized as the *Tortellino*'s ancestors, it is not until the sixteenth century that we find a dish very similar to the present-day recipe, described in the Epulario by Giovanni Rosselli [Rosselli 1516]. However, until the nineteenth century Tortellini continued to be served during festivities and to be confined to the tables of the richest people in Bologna. As of today, instead, to witness the fame of this special type of pasta, the *Tortellino* is well known to the point that it is regularly enjoyed during the meals on many tables around the world.

During all these centuries, the main mechanism that has permitted to pass on the tradition of preparing Tortellini has been the interaction between expert people. The *Sfogline* (i.e., the female cooks who are specialized in preparing Tortellini) taught their apprentices the secrets of their art and, in this way, their knowledge was kept alive, being transmitted from generation to generation.

One of the questions this paper answers is whether this apprenticeship process can be improved or further supported, based on the use of modern digital communication technologies. Far from thinking of substituting a traditional methodology that has undoubtedly proved to be successful during these past few centuries, we provide here a little contribution in support to the idea that this special kind of enactive knowledge (i.e., the knowledge of how to do very practical things and artifacts) can be made widely available to newcomers, also with the beneficial support of digital technology [Shimizu *et al*. 2006]. Indeed, the most recent gesture following and recognition techniques may play a central role in supporting our claim [Liu and Kavakli 2010].

There is no doubt, in fact, that Nintendo Wii, Microsoft Kinect and Sony Move have revolutionized the way humans can interact with computers for leisure and educational purposes. While advances in such field have been certainly available for long, before their commercial launch, in many research labs around the world, their special achievement resides in being able to bring such technologies to the general public. Slowly, but at a pace, people are understanding that it is possible to

interact with technology-augmented objects, while performing intuitive and natural hand gestures, as simple as those John Anderton (i.e., the character impersonated by Tom Cruise) does in the *Minority Report* blockbuster movie.

While the voice and gesture recognition technologies have certainly shown their ability to strongly influence the launch of new gaming consoles in the computer gaming industry, their opportunities still need to be widely explored in other fields. One of the sectors that could certainly benefit from supporting richer and more complex interactions between computers and humans is that of technical and artistic exhibition fields. Indeed, exhibitions, places where usually something amusing, interesting and educational can be enjoyed, often rely on digital multimedia (e.g., computer graphics, audio, video, etc.) to convey more information to their visitors. However, such technologies are usually utilized to simply display information, or even interact using very poor interaction schemes. For this reason, a wealth of research is working on making new steps in the direction of providing all those who visit an exhibition with a deeper and richer interaction with the delivered contents, as in this way visitors are given an opportunity to learn/enjoy more than what can be simply seen or heard using traditional systems.

In such context, we want to describe our experience in designing and implementing a multimedia system for the City of Bologna, in Italy, that will be displaying at the 2010 Shanghai World Expo. Making a little digression to explain the context of our work, the City of Bologna has been selected among hundreds of cities as a “model of virtuous practices” in terms of creativity, technological innovation, urban transformation and social inclusion. Along with other fifty-nine cities selected from all around the world, Bologna has been granted a pavilion in the area dedicated to the theme “Better City, Better Life”. In such scenario, with the purpose of exposing one of the masterpieces of the Italian culinary heritage that was devised in Bologna, we designed and implemented an interactive system that provides its visitors with a richer experience, whose duration in time amounts to a few minutes, on how the traditional pasta cooked in Bologna, the *Tortellino*, is prepared from water, raw flour, eggs and its stuffing ingredients.

Instead of simply basing our system on a passive video and audio narration, we decided to design a sort of interactive storytelling mechanism where each visitor stands at a specific location on stage and watches/listens to, on a nearby screen in front of him/her, an explanation of the actions required to complete each given phase of the preparation of the recipe. A video camera standing above records the movements the

visitor performs to imitate the delivered instructions, while feeding them to a software module, which interprets them and checks for their culinary correctness. In real-time, the user sees in return the system displaying a stop-motion animation of his/her actions with its outcome. In the case user's movements are correct, the system proceeds displaying the next phase of preparation of the recipe. Nonetheless, a user that has never cooked before the Tortellini may have some difficulties to immediately grasp the correct actions that should be accomplished; for this reason our system also handles the case where the movements are wrong, asking the user to perform again the correct set of cooking actions related to the current phase of preparation of the recipe.

The aforementioned functionalities are implemented using three main software modules: a hand follower, a gesture recognizer and an audiovisual feedback displayer. Within each phase of making a *Tortellino*, the following interactions between our system and a user are performed:

a) in return to the instructions delivered to the user on how to perform a certain cooking action, as displayed on a monitor by a *Sfoglina* that accomplishes the correct sequence of movements, the user mimics the actions, moving freely his/her hands in the air,

b) a **hand follower** tracks all user's movements while displaying their real-time digital representation on a monitor in front of the user,

c) a **gesture recognizer**, then, checks whether user's gestures are either correct or not, and finally,

d) an **audiovisual feedback displayer** shows a stop-motion animation of the effects that the cooking actions of the user have on the pasta he/she is manipulating (in the case where the user performed the correct steps). Instead, a message asking the user to repeat previous actions is delivered to the user, in the case when he/she made any mistake.

Generally speaking, the main scientific challenges we had to face with to implement such a system have been those of devising the most appropriate techniques and algorithms able to allow an automatic system to recognize a predefined set of movements (as those performed by a real cook), without the help of any other hardware device, but a simple camera. In particular, in response to the problems raised by items b) and c) above, we designed and implemented both a hand follower and a gesture recognizer which were able to track, and then to interpret, the gestures each given visitor would have made, while trying to mimic a real cook.

The main contribution of this paper is to demonstrate that our hand follower and gesture recognizer are able to perform well during real exhibitions, even with non-expert visitors, under those strict timing and performance requirements that real exhibitions usually pose.

As per the discussion about the design of the audiovisual feedback displayer (and also of the mechanism in charge of delivering stop-motion animations back to visitors), we have to admit that it is out of the scope of this paper. The motivation behind this choice is that this component of the system, although important, did not present any particularly intriguing challenge from a scientific viewpoint, and was therefore outsourced to an external media agency based in Bologna (whose name is Articolture) that built it, in isolation. Upon realization of all the three modules, they were finally assembled together for their final use.

The remainder of this paper is organized as follows. In Section II, we give an overview of the technical and scientific research that falls closest in scope and methods to our work. A step-by-step explanation on how our system works is given, in Section III, based on an artistic hand-drawn storyboard we developed for this aim. Sections IV and V are devoted to a deeper discussion on the technical design aspects concerning our hand follower and gesture recognizer, respectively. We finally conclude the paper with Section VI.

2. RELATED WORK

A wealth of research has been carried out in the past few years in the area of advanced gesture-based human-computer interaction interfaces. However, we keep our discussion confined to the research results that fall closest in scope and in technological aspects to ours. In other words, among the many possible alternatives, we focus here on those that exploited advanced digital recognition technologies to be used for artistic and exhibitive performances, especially [Bevilacqua *et al.* 2001; De Ponti *et al*. 2009; Ferretti and Roccetti 2005; Ferretti *et al*. 2005; Ferretti *et al*. 2008; Ferretti *et al*., 2009; Kasap *et al*. 2009].

A first prominent example along this line is the following. In [Carrozzino *et al*. 2008], authors describe their implementation of the Virtual Museum of Sculpture in Pietrasanta in Italy, a technologically advanced museum where a user can physically interact, touching and watching the 3D digital models of certain sculptures. Such implementation, performed as a part of the research carried out at the S. Anna PERCRO Lab in Pisa, Italy, is based on the use of haptic interfaces which, utilizing force feedback technologies, give visitors the impression of touching an artwork that does not exist in the reality.

A first consideration about this work is that a system like the one mentioned above requires the use of sophisticated hardware transducers and devices, which clearly increases its cost, and this may count negatively in some given context. Further, although interesting, this type

of technologies fall far from our idea of “pure” form of interaction which requires the system to understand and to interpret what the user is doing, without the help of external hardware devices or specialized sensors. In fact, we feel that the more the user will be able to move and perform hands-free, the more he/she will enjoy the artistic performance, supposing the technology is playing only a minor role in what is happening.

More similar to ours is an interesting experience conducted at IRCAM, concerning the process of learning music theory, as described in [Bevilacqua *et al*. 2010]. In brief, this system is comprised of both hardware (accelerometers and gyroscopes) and software (a gesture following and recognition system) components, and has the aim of providing a technological aid to teach an apprentice how a music orchestra can be conducted. When an apprentice in conducting music performs the gesture related to a piece of music to be played, this system checks whether that gesture is close enough to what the teacher previously taught, as it was coded within the system. Again, the interactions that this system supports strongly rest upon the use of specific hardware and sensors, while our aim is to manage all the interactions with non-specialized devices, like a simple camera, moving all the intelligence towards the software side of the system.

Techniques for gesture following have also been explored to aid people learning and appreciating an intangible form of art such as that of dancing [Magnenat-Thalmann *et al*. 2007]. In that paper, researchers of MIRALAB research center presented a system designed to provide a learning framework for folk dances. In particular, they used an optical motion capture system to record the movements performed by professional dancers, with the aim of building a digital model of a given folk dance. Apprentices then use a web interface to interact with that digital model, with the aim of improving their comprehension of how that dance can be played. Interestingly, the authors show how their system effectively helps in this process, as their experimental results show students learn with a steeper learning curve, on average.

Many similarities may be found between this and our work, as both approaches make in some sense a “pedagogical” use of advanced digital technologies. However, the system discussed in [Magnenat-Thalmann *et al.* 2007] does not provide any interactive feedback on how an apprentice performs his/her action after he has learnt them; rather it supplies a model that can only be run and rerun to better understand how dancers carry out any single movement.

3. HOW TO PREPARE A TORTELLINO: PLAYING WITH A GESTURE RECOGNIZER

How our interactive system accompanies a user while performing each cooking phase of the preparation of a *Tortellino* is summarized in Figure 1. Starting from the topmost action box, first a video clip, displaying a *Sfoglina* while cooking, explains what should be done at that given phase of preparation of a *Tortellino*. Then the user is asked to imitate the actions that have been shown. While he/she performs, his/her hands are tracked and reproduced on the display (second action box in Figure 1). On action, user's movements are recorded by a webcam and then passed to our gesture follower and recognizer system that returns either a positive or a negative feedback, depending on how accurately the user has mimicked the correct action set. In the positive case, a stop-motion animation representing the effects that the actions of the user have on the pasta he/she is cooking is displayed on the monitor. In the negative case, the user is given a second chance to better perform his/her cooking actions.

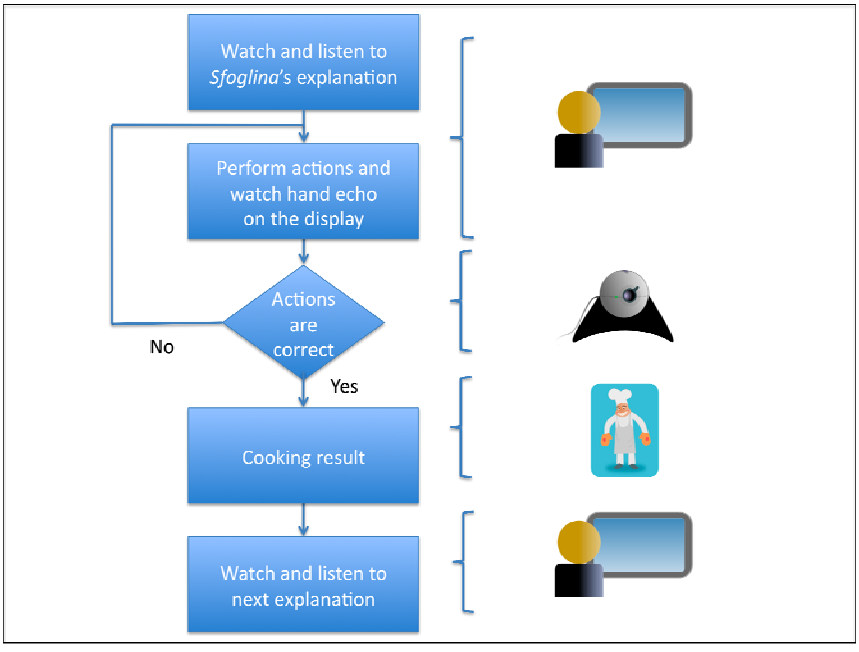

Fig. 1. Playing with the Tortellini interactive system.

To better understand how a *Tortellino* is prepared, we now proceed explaining each and any phase of its preparation, using a hand drawn storyboard we developed to this aim. The storyboard illustrates what a user exactly sees when interacting with our system. It accurately depicts every single element displayed by the mechanism in return to the actions performed by a user. Needless to say, this tool has represented a very valuable means that guided us in the design and development of our system. Let us start with the narration.

At the very beginning, a user watches an introductory video where a real cook (a *Sfoglina*), talking from a traditional kitchen, welcomes the user and explains the role that the Tortellini have played in the history of Italian cuisine (Figure 2). In this phase, our cook also explains that the user is going to take a trip through an interactive experience whose result will be that of becoming acquainted with the main steps required in preparing such a special kind of Italian Pasta.

Fig. 2. *Sfoglina* explaining.

Figure 3 shows the first three interactive phases. Starting from a situation where the cooking board is empty (phase 1), we then move to phase 2 where a box of flour appears on the right of the cooking board. The virtual *Sfoglina* then asks the user to take some and place it at the center of the board. At this point, the user may try to imitate the cook putting the flour at the center of the board. If this is successfully accomplished, and recognized as such by our system, we move to phase 3, where the user is asked by the *Sfoglina* to perform again an action similar

to the previous one: that of taking the eggs, which in the meantime appeared on the right of the board, and placing them at the center of the board.

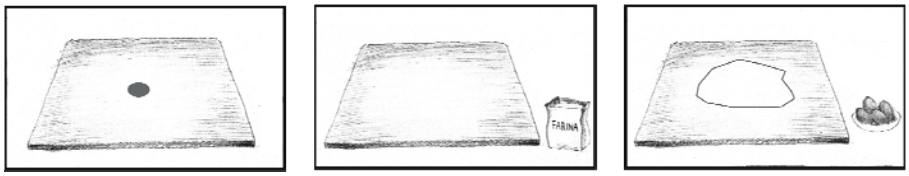

Fig. 3. Left to right: phases 1, 2 and 3.

Phase 4 is comprised of a stop-motion animation showing the flour and eggs added at the center of the cooking board, waiting to be mixed (leftmost drawing of Figure 4). At this point, the *Sfoglina* asks the player to mix the ingredients, performing a few circular movements to knead the flour and eggs at the center of the board. If the user correctly performs this action, as recognized by our system, a ball of dough appears (phase 5) placed at the center of the board (central drawing of Figure 4), and the *Sfoglina* congratulates the user with a “well done, my friend!”. During phase 6, instead, a rolling pin becomes visible, while the *Sfoglina* asks the user to place his/her hands at the ends of the pin, thus getting ready to spread the dough.

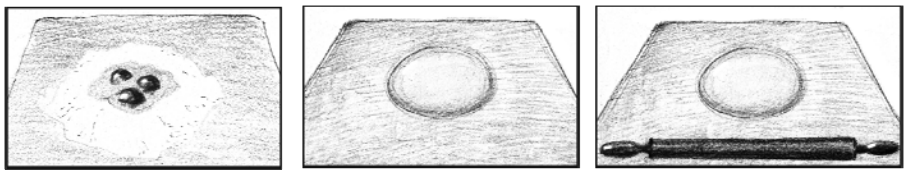

Fig. 4. Left to right: phases 4, 5 and 6.

The user can now imitate the *Sfoglina* and spread the dough, repeatedly moving the rolling pin, moving forward and backward his/her hands, freely in the air. Once this has been successfully done and recognized by our system for at least three times, (phases 7 and 8), the *Sfoglina* congratulates the player on correctly performing these steps and the game moves to phase 9 (rightmost drawing in Figure 5), where a thin foil of dough lies at the center of the cooking board, as a result of the previous spreading actions.

Phase 10 (leftmost drawing in Figure 6) begins with a pastry cutting wheel appearing, lying on the pasta foil, while the *Sfoglina* instructs the

user to grasp it and cut the foil of dough, through two subsequent sets of movements.

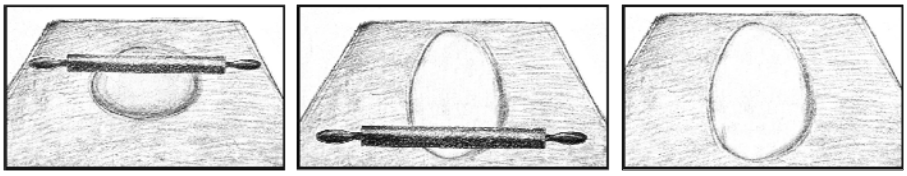

Fig. 5. Left to right: phases 7, 8 and 9.

The first set of movements requires the player to cut the foil with a top-to-bottom movement (phase 11) to be repeated for three consecutive times, thus yielding three parallel lines in the dough (central drawing in Figure 6). The second set of movements, instead, entails that the player cuts the dough along three parallel lines, with a left-to-right gesture. If all this is done well, a stop-motion animation is presented displaying the echo of the movements of the user and its final result: four squares of dough lying on the board (phase 12, rightmost drawing in Figure 6).

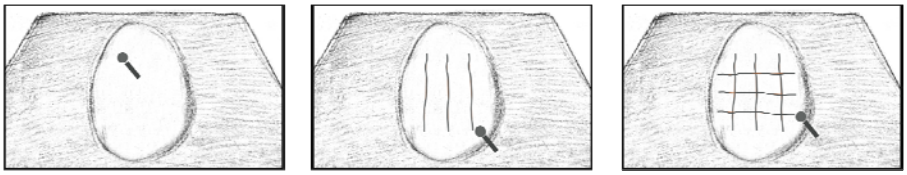

Fig. 6. Left to right: phases 10, 11 and 12.

Phase 13 begins with a zoom on three adjacent squares of dough, as well as with the appearance of the stuffing at the bottom right side of the screen (leftmost drawing in Figure 7a). The user is now instructed to add some of the stuffing on each of the three squares. Phase 14 (central drawing in Figure 7a) represents the player trying to perform the stuffing action. If this is performed well for all the times it is required, the game moves to phase 15 (rightmost drawing in Figure 7a), with each square of dough perfectly stuffed.

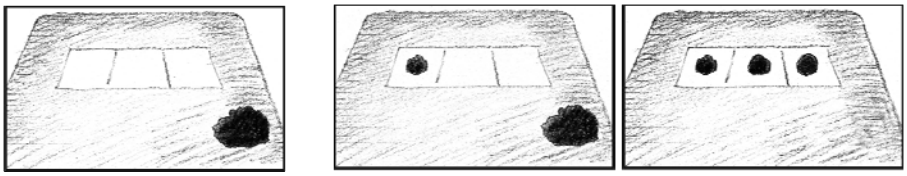

Fig. 7a. Left to right: phases 13, 14 and 15.

Clearly, the player is at this point ready to close the dough and get the final *Tortellino*. The three final phases of the preparation process (phases 16, 17 and 18) begin with phase 16, where a single stuffed square of dough is displayed at the center of the board. The *Sfoglina* instructs the player to grasp the two opposite ends of the dumpling and to close it. In the positive case, this action brings to a halfway closed *Tortellino*, as displayed in the central drawing of Figure 7 (phase 17). The very final action the *Sfoglina* asks the player to perform is to keep still the leftmost of the other two opposite ends, while sealing the rightmost one on it. The final result of this action is a final *Tortellino*, as shown in phase 18 (rightmost drawing in Figure 7b). At this point, the Tortellini only lack to be boiled in salted water, for just the appropriate amount of time, but we have omitted this final step as this is not intriguing enough from a hands-free manipulation standpoint. We can then conclude this Section with the *Sfoglina* (Figure 8) thanking the user, while inviting him/her to join her in Bologna, to taste the real *Tortellini* pasta.

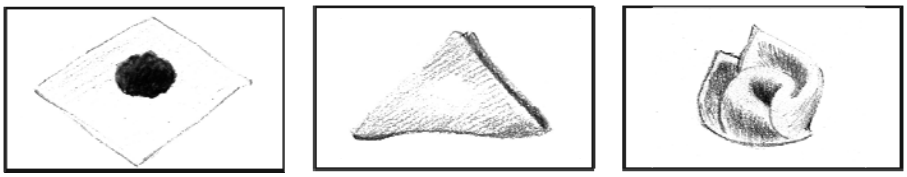

Fig. 7b. Left to right: phases 16, 17 and 18.

Fig. 8. Final closure.

4. HAND FOLLOWER: IMPLEMENTATION AND DISCUSSION

As already mentioned, our system is composed of three modules: i) a hand follower, ii) a gesture recognizer, and iii) an audiovisual feedback displayer. In this Section, we will focus on how the hand follower module has been designed and implemented. This was a very complex design step that led us to explore the implementation of a set of alternative methods (and of corresponding algorithms) that could best fit with our realistic setting. In particular, we concentrated on three different alternative approaches.

The first approach we took into consideration was that based on the use of a Wii remote [Lee 2008], the second one was that developed based on the use of the OpenCV software library [Manresa *et al*. 2005], and finally the third one was devised to implement a gesture following technique we developed at the University of Bologna, without resorting to any external software library [Semeraro 2008].

The motivation for exploring the Wii technology is that it represents a natural candidate for this type of systems, as it can be simply turned into a motion-tracking system by having a user wearing an infrared emitting LED, thus providing a way to estimate the x-y position of the user's hands on the screen, as well as their relative distance [Lee 2008]. This technique has been thoroughly studied in literature and extensively used in practice, but in our case has one easily identifiable disadvantage. While, in fact, the Wii technology is mature, stable and simple enough to be proficiently used in practice (almost as an off-the-shelf software component), nonetheless using it in our scenario would mean equipping each user with gloves (or similar wearable objects) capable of emitting an infrared light. Unfortunately, this would have represented a major obstacle for our system to be deployed at the Shanghai World Expo. In fact, as we expect several thousand visitors spending time at our stand to enjoy our exhibition, this would entail wearing and removing gloves for each of them, with an unbearable waste of time. Not to mention the fact that wearable objects also represent a source of possible problems (e.g., gloves may get lost or be damaged). For all the aforementioned reasons, after experimenting with this technology, we dropped this alternative as not appropriate to our case.

As a second alternative we considered using the OpenCV library. It provides a set of functions that can be useful to track the hands of a person while moving. In particular, such libraries provide algorithms that identify the pixels whose color is closest to a given pigment [Manresa *et al*. 2005]. Using such libraries, it is possible to identify the pixels of the hands of a given user, verifying which pixels take values between two color

thresholds, thus identifying the range of colors of the skin of those given hands. Once this information is available, it is possible to identify the center of mass of both hands, by simply averaging out the x-y position of the points that compose their contours.

Unfortunately, there is a first problem with this technique we were compelled to modify. In fact, the aforementioned procedure could bring some problems, as this function could provide an incorrect estimate of the center of mass of a user's hands, in case also the arms, for example, or other parts of the human body, were erroneously detected.

For this reason, we devised a new approach, based on the use of the OpenCV library, which simply finds the position of the farthest ahead point of each hand, and accordingly we developed an algorithm which is sketched in Figure 9.

1: *colorRange* ← user's skin color acquisition.
2: *A(t, colorRange)* ← Skin pixels in the left hand area at time t;
3: *B(t, colorRange)* ← Skin pixels in the right hand area at time t;
4: *leftHandY(t)* ← $\max_{Y} A(t, colorRange)$;
5: *leftHandX(t)* ← x s.t. $y = \max_{Y} A(t, colorRange)$;
6: *rightHandY(t)* ← $\max_{Y} B(t, colorRange)$;
7: *rightHandX(t)* ← x s.t. $y = \max_{Y} B(t, colorRange)$;

Fig. 9. OpenCV-based hand follower algorithm.

Since the video cameras are placed above the head of each player, the modification we introduced to the OpenCV approach (precisely, lines 4 to 7 in the algorithm of Figure 9) guarantees that the position of each hand is precisely identified.

Clearly, a great advantage of this OpenCV-based approach is that users do not need any device to interact with the system. Although promising, we contrasted OpenCV against a third alternative we developed based on the idea that no external software library should be needed. Indeed, a wealth of literature, discussing the fact that the amount of operations required to track user's hands with the OpenCV library could be in principle very processor-intensive, led us to explore this new way.

We therefore devised a third custom (home-made) method. This, as the preceding one, does not require users wearing any sensor, while at the same time implements a much faster tracking algorithm that can be used for our scope. The hand follower algorithm we devised works as follows and is summarized in Figure 10.

In essence, to recognize a hand, it compares subsequent frames taken by a camera. Suppose, for example, we want to locate the hands of a given

user. At the very beginning, before the user approaches the stage, the camera takes a picture of the background and stores it as a reference frame (line 1, algorithm of Figure 10). As soon as the hands of the user get into the webcam's sight, subsequent frames are taken and then compared with the reference frame, using a function subtracting on a per-pixel basis (lines 2-3). If the difference between the current frame and the reference background exceeds a given threshold value, for an area larger than that considered of minimum size (e.g., the size of a kid's hand), then there is a positive match, and the hand is identified and hence tracked (lines 4 to 7 in the algorithm of Figure 10).

Also such home-made approach has advantages and disadvantages. The most prominent advantage is that it is very simple and fast to perform, while, intuitively, the main drawback is that it seems to rely on a coarser tracking methodology, more easily subject to interferences and prone to giving false positives, as well. For instance, each object passing through the area of interest during the tracking process can give rise to a positive match (that is, a false positive).

```
1: background ← background color acquisition;
2: A(t, background) ← subtract background from current frame in the left hand area at time t;
3: B(t, background) ← subtract background from current frame in the right hand area at time t;
4: For each (x,y) s.t. A(t, background) > threshold
5:     left-hand(x,y) ← detected;
6: For each (x,y) s.t. B(t, background) > threshold
7:     right-hand(x,y) ← detected;
```

Fig. 10. Custom hand follower algorithm.

As a courtesy towards the reader, we have summarized the main pros/cons of each of the three considered approaches (Wii, OpenCV and custom) in Table I.

Table I. Hand follower approaches: advantages and disadvantages.

Technology	**Positive**	**Negative**
Wii	Fast/Available	Hardware needed
OpenCV	Hands free	Complexity
Custom	Hands free	Accuracy

Rather, it is important to point out now that to take our final decision we carried out an extensive set of experiments (more than one hundred)

with different users and diverse conditions of illumination of the stage. A summary of the most important results we got is the following.

Both the OpenCV-based and the custom approaches were able to rapidly track the hands of a given user, yielding a responsiveness delay (that is the time between the moment when the user moves his hands and the moment when the system tracks and echoes them on the display) of a few tens of milliseconds, on average.

Rather surprisingly, instead, results were controversial as to the accuracy, essentially depending on the type of camera to be used.

In particular, when a very cheap camera with basic recording functions (e.g., a web camera) was employed, the ability of the OpenCV-based method in exactly identifying the hands was almost near to 100% of cases, while the performances of our custom method dropped below the threshold of 80%, on average. This situation was surprisingly inverted in the case when a professional camera with advanced functionalities was exploited to take subsequent frames. In this case, the custom method outperformed the OpenCV-based one, yielding nearly the 90% of positive matches. In these experiments, indeed, the OpenCV-based performance could drop down to 80%, its performances strongly depending on the changes of the illumination on the stage.

The motivation behind all this is as follows. While cheaper cameras usually come with automatic self-tuning mechanisms that automatically regulate all lighting parameters, more sophisticated video cameras come instead with a set of advanced functions that an expert user should adjust, on a per single-case basis, to get the best performance from the camera. Hence, while, for example, with a self-stabilizing camera all shadowing effects due to illumination are automatically filtered out, instead the various lighting functions of a sophisticated camera need to be adequately tuned to achieve a similar result. In conclusion, when we experimented with a basic self-regulating camera, the OpenCV-based method outperformed, on average, our custom mechanism, as the camera automatically eliminated all those shadowing interferences towards which OpenCV is more sensible (as it employs a finer tracking method). Instead, when an advanced camera was used without any particular attention to its setting, the OpenCV method became more sensible to interferences due to changes in illumination, while our custom method performed better, as it was developed based on techniques which are coarser, but more robust in some sense, and hence less prone to interferences due to illumination changes.

All this has given us the following precious information. When in Shanghai, if a sophisticated camera will be used (as it is plausible), then,

either the camera is to be perfectly adjusted based on the illumination settings (and in this case it is worth using the OpenCV-based method) or, in the negative case, our custom method is to be preferred.

5. GESTURE RECOGNIZER: IMPLEMENTATION AND DISCUSSION

Upon identification of hands, then the problem becomes that of recognizing a movement as either correct or not. We tried two different alternatives to implement an effective gesture recognizer for assisting an apprentice in the cooking of Tortellini: a first one that implemented an algorithm we devised to recognize hand movements based on the fact that given areas in the space are traversed when a gesture is performed, and a second one that utilized a slightly modified version of the well known gesture recognition system, termed *$1* [Wobbrock *et al*. 2007].

As to the first method, upon designing our gesture recognizer, we started by analyzing the predefined set of gestures as described in our storyboard, thus realizing that all the various phases of preparation of the Tortellini (except for phase 4, which we will discuss on a separate basis) were characterized by the fact that each hand movement begins in an initial area and ends into a final area, following a given trajectory connecting the two areas. In addition, the action of following a given trajectory must be done within a given time period, after which that movement has been performed too slowly to be considered correct.

Not only, a user may also begin his cooking action, and then immediately leave for diverse reasons. Hence, in order to avoid having the system stuck waiting for some action that will be never done, we had also to handle such type of occurrences by means of a timeout.

All these considerations represent the rationale behind the implementation of our custom gesture recognizing algorithm, which is sketched in Figure 11.

```
1:  if (position.isInInitialArea()) {
2:      position ← currentPosition();
3:      startTimer();
4:      while (position.isNotInFinalArea()) {
5:          if (position.leavesCorrectTrajectory() or timeout) {
6:              error();
7:          }
8:      }
9:      success();
10: }
```

Fig. 11. Custom gesture recognizer algorithm.

The algorithm of Figure 11 does not need any further explanation, but that each trajectory is recognized as correct if it flows within a stripe of a given size (the idea is that a certain degree of tolerance is admitted). This mechanism was devised to make our recognizer able to consider as correct a wider set of movements with slightly different trajectories that differentiate only on the basis of a few geometric differences with respect to the basic one.

As already anticipated, phase 4 of the preparation of the Tortellini needs a different approach. In fact, in this phase, the user performs a circular movement with his/her hand (mixing the ingredients) at the center of the cooking board. For this reason, we devised the following original solution. In essence, the board was split into two different parts, divided by a central axis. A mixing movement is hence recognized as correct if the hands of a given user traverse the central axis of the board, back and forth for a given number of times. Figure 12 reports a sketch of this algorithm.

```
1:  x ← central axis;
2:  N ← number of required axis traversals;
3:  side ← the position is on the left or right side of the x central axis;
4:  i ← traversal counter, initially set to 0;
5:  previousSide ← currentSide();
6:  while (i < N) {
7:          side ← currentSide();
8:          if (side == right and previousSide == left) {
9:              previousSide ← right;
10:             i++;
11:         }
12:         if (side == left and previousSide == right) {
13:             previousSide ← left;
14:             i++;
15:         }
16: }
```

Fig. 12. Custom gesture recognizer algorithm (for phase 4).

As an alternative to the two custom algorithms we have just presented, we also took into consideration the idea of exploiting a modified version of the well known *$1* algorithm [Wobbrock et al. 2007]. In essence, *$1* exploits a library comprising a set of graphical patterns to be contrasted against the trajectories produced by a given movement. Upon recognizing a user gesture, *$1* returns a score between 0 and 1, where higher scores correspond to closer matches between the trajectory of a given movement and a given pre-recorded curve.

Again, we modified and adapted the *$1* algorithm to our specific case, because it recognizes as similar gestures that instead should be considered

different, according to the rationale of our system. There is an obvious motivation for this "abnormal behavior" of *$1* and is due to the fact that it considers as similar two gestures, which can be instead altered by a translation or an expansion or a rotation transformation. Instead, our system to work correctly cannot tolerate transformational invariants, as *$1* does. Just as an instance of this, consider the character "8" and the symbol "∞", which clearly differ in terms of both size and inclination. They would be identified as similar by *$1*, while we could not tolerate this if these were trajectories produced by the movements of a given user.

To provide an example of the type of modifications we had to introduce into *$1* to make it answer to our needs, take again the case of mixing the ingredients with a circular movement. In this case, our modification to *$1* has consisted in binding the correct movement to be recognized as performed within a specific area, i.e., a square that lies at the center of the cooking board. Thus, when mixing the ingredients a user must move his hand describing an "O" within this square to perform the correct movement. All those movements, recognized as "O", but executed outside that given area, are not recognized as correct, based on our modification.

Interestingly, after implementing both the solutions (our custom solution and that based on *$1*), we found that ours performed better in recognizing all the cooking gestures we were interested in. The reasons, mainly three, are easily explained by following the usual example of ingredients mixing.

The first reason is that many people interpret the action of mixing the eggs and flour moving back and forth their hand, along a horizontal shape that is much closer to an ellipse than to a circle. So, what should we recognize here: a circle, an ellipse or both? One could think that this problem could be easily solved either by lowering the similarity threshold of *$1*, or introducing a larger set of allowed curves, including circles, ellipses and even straight lines. Unfortunately, all this has the negative side effect that many various movements could then give rise to a positive match, thus jeopardizing the efficacy and the efficiency of the gesture recognizer. The second reason is that confining the area of movement into a square, even if needed if we want to use *$1* in our case, gives many additional mistakes, as it is sufficient that the user falls with his hand out of the given area to cause an error. The third and final reason is that while our custom gesture recognition algorithm can easily work with both the two hand follower modules we have described in the previous Section, for clear reasons it is much harder to get the *$1* gesture recognizer module working well with the custom hand follower that exploits the pixel difference between the background and the current frame. This

inconvenience is easy to understand and is caused by the fact that our custom hand follower recognizes hands in terms of a given area (subject to a relevant difference in pixel colors), while the *$1* algorithm analyzes gestures described by single points on a plane. Hence the incompatibility follows.

It is worth to conclude this Section by mentioning that, after having implemented and extensively subjected to testing both solutions (our custom one and the *$1* one), our final determination was that of using our custom method. All the experiments we carried out, in fact, confirmed on the field the convincing motivations we have already discussed.

6. CONCLUSION

We designed and implemented a multimedia system that simulates the process of preparing one of the masterpieces of the culinary heritage of the Italian city of Bologna, the *Tortellino*. With our system, anyone is able to enjoy a hands-free, virtual experience focused on how to make a *Tortellino*, while mimicking the movements a real cook would perform to prepare its recipe. The main scientific contribution of our work has been that of designing and implementing effective software mechanisms able to recognize the actions an apprentice in cooking would perform to prepare the Tortellini pasta. Witnessing the importance of the results we got, our multimedia system has been selected to be part of the expositions that will represent the City of Bologna at the Shanghai Universal Expo, in October 2010.

7. ACKNOWLEDGEMENTS

We feel indebted towards the FIRB Damasco Project providing financial support to our work, and to our colleagues L. Donatiello, R. Grandi and A. Varni who selected our system to represent the City of Bologna at the Shanghai Expo. We express our gratitude to all our undergraduate students (CS Class: Multimedia Systems and Applications, University of Bologna). Their names follow: Beatrice Bacelli, Cristian Bertuccioli, Carlo Brualdi, Antonio Casamassima, Giovanni De Marco, Andrea Di Toro, Luca Leoni, Andrea Marcomini, Giacomo Giorgi, Mirko Pedrini and Diego Rodriguez. They assisted us during the implementation of the system. Special thanks are devoted to Articolture, a Bologna-based media agency, which implemented the audiovisual feedback displayer. Finally, M. Roccetti and M. Zanichelli also thank Violo for her precious suggestions and comments on the subject, which our system touches upon.

REFERENCES

BEVILACQUA, F., ZAMBORLIN, B., SYPNIEWSKI, A., SCHNELL, N. GUÉDY, F. AND RASAMIMANANA, N., 2010. Continuous realtime gesture following and recognition, In *Proceedings of International Workshop on Gesture in Embodied Communication and Human-Computer Interaction*, Lecture Notes in Artificial Intelligence 5934, Springer, Berlin Germany, 73-84.

BEVILACQUA, F., NAUGLE, L. and I. VALVERDE, I., 2001. Virtual dance and music environment using motion capture. In *Proceedings of 2001 IEEE Multimedia Technology and Applications Conference*, Irvine CA, 1-8.

CAPATTI, A., MONTANARI, M., O'HEALY, A., 2003. Italian Cuisine: A Cultural History, *Columbia University Press*, New York.

CARROZZINO, M., EVANGELISTA, C., SCUCCES, A., TECCHIA, F., TENNIRELLI, G. AND BERGAMASCO, M., 2008. The virtual museum of sculpture. In *Proceedings of 3rd ACM International Conference on Digital interactive Media in Entertainment and Arts*, Athens, Greece, 100-106.

DEPONTI, D., MAGGIORINI, D., AND PALAZZI, C.E., 2009. DroidGlove: an android-based application for wrist rehabilitation. *In Proceedings of IEEE International Conference on Ultramodern Telecommunications*, St. Petersburg, Russia, 1-5.

FERRETTI, S. AND ROCCETTI, M., 2005. Fast delivery of game events with an optimistic synchronization algorithms in massive multiplayer online games, In *Proceedings of 2nd ACM International Conference on Advances in Computer Entertainment Technology*, Valencia, Spain, 405-412.

FERRETTI, S., PALAZZI, C. E., CACCIAGUERRA, S. AND ROCCETTI, M., 2005. A RIO-like technique for interactivity loss avoidance in fast-paced multiplayer online games, *Computers in Entertainment*, 3(2), 1-11.

FERRETTI, S., ROCCETTI, M. AND STROZZI, F., 2008. On developing tangible interfaces for video streaming control: a real case study, In *Proceedings of 18th ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Braunschweig, Germany, 51-56.

FERRETTI, S., ROCCETTI, M. AND ZAMBORLIN, B., 2009. On SPAWC: discussion on a musical signal parser and well-formed composer, In *Proceedings of 6th IEEE Communications and Networking Conference*, Las Vegas NV, 1-6.

KASAP, M. CHADHURI, P. AND MAGNENAT-THALMANN, N., 2009. Fast EMG-data driven skin deformation. *Computer Animation and Virtual Worlds Journal*, 20(2-3), J. Wiley & S., 153-161.

LIU, J. AND KAVAKLI, M., 2010. A survey of speech-hand gesture recognition for the development of multimodal interfaces in computer games, In *Proceedings of 2010 IEEE International Conference on Multimedia and Expo*, Singapore, 1-5.

MAGNENAT-THALMANN, N., PROTOPSALTOU, D., AND KAVAKLI, E., 2007. Learning how to dance using a web 3D platform, In *Proceedings of 6th International Conference on Web-based Learning*, Edinburgh, UK, 1-12.

ROSSELLI DE', G., 1516. Epulario: qual tratta del modo di cucinare ogni carne, ucelli, pesci d'ogni sorte, Niccolò Zoppino e Vincenzo di Paolo Pub., Venice Italy.

SEMERARO, A., 2008. Minority report: a case study, The *Made in UNIBO* Exhibition, Museum of Modern Art, Bologna.

SHIMIZU, N., KOIZUMI, N., SUGIMOTO, M., NII, H., SEKIGUCHI, D., AND INAMI, M., 2006. A teddy-bear-based robotic user interface, *Computers in Entertainment*, 4(3), 8-13.

WOBBROCK, J. O., WILSON, A. D., AND LI, Y. 2007. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *Proceedings of 20th Annual ACM Symposium on User interface Software and Technology*, Newport RI, 159-168.