AN ACCESSIBLE APPLICATION TO FOSTER FIGURE SKATING SKILLS USING GAMIFICATION

Yuxi Xiao[1], Joshua Lai[2]

[1]University High School, 4771 Campus Dr, Irvine, CA 92612
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

The world of figure skating involves more than meets the eye, with precise and difficult moves judged subjectively, leading to biases and cultural influences impacting scoring. Skaters face physical and mental challenges, including pressure to conform to beauty standards. Addressing these issues, a proposed skating companion app aims to democratize figure skating by providing affordable, engaging, and safe training. Recognizing financial constraints, the app offers lessons, tips, and interactive elements to assist learning. It introduces a gamification element for simulated competition experiences, creating a risk-free environment for younger skaters during their formative years. The app aspires to be a catalyst for positive change, promoting inclusivity, safety, and education within the figure skating community, empowering individuals to navigate the complexities of this captivating sport while fostering a supportive and inclusive environment.

KEYWORDS

Figure Skating, Gamification, Simulation, Training

1. INTRODUCTION

At a glance, the world of figure skating seems relatively straightforward. Skaters practice a dance on the ice, perform said dance, and are judged based on how good the performance was. However, there is so much more to what constitutes a "good performance" than what meets the naked eye. Executing the moves required in figure skating is incredibly precise and difficult [2]. And skill alone is often not the only factor. The sport's subjective nature introduces an additional layer of complexity. Judges assess not only technical proficiency but also the artistic interpretation of the program. This opens the door to interpretation bias, where personal preferences and cultural influencesmay impact scoring [7][8]. Skaters must navigate these nuances, adapting their performances to align with the subjective inclinations of the judging panel, further emphasizing the intricate dance between athleticism and artistry [14]. Such biases extend beyond the rink, and often bleed into news reporting [9]. As for the journey of a skater before they even begin computing, training begins very early in childhood, raising questions as to the morality of such intense physical exercise for young minors [11][15]. Not everybody who wishes to skate can afford training at such a young age, nor may they want to be subject to its brutality. Moreover, the competitive world of figure skating is not only physically demanding but also mentally and emotionally taxing. Skaters face immense pressure to conform to rigid standards of beauty and athleticism, which can lead to issues of body image and self-esteem [3]. The emphasis on perfection in jumps, spins, and artistic expression places a heavy burden on these athletes, often pushing them to the limits of their physical and mental endurance.

David C. Wyld et al. (Eds): NBIoT, MLCL, NMCO, ARIN, CSITA, ISPR, NATAP-2024
pp. 175-185, 2024. CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.140515

EJ Kovacs et al.'s study investigates the impact of two training programs on figure skaters' postural control. One program emphasizes neuromuscular training, while the other focuses on conventional off-ice training. Through a randomized trial, the research concludes that off-ice neuromuscular training is more effective in enhancing figure skaters' postural control, providing valuable insights for future training protocols. Eb, Jeroen van der, Sjoerd Gereats, and Arno Knobbe introduced SkateView, a device for elite speed skaters with precise training measurements. Our skating app, inspired by this technology, targets all skating enthusiasts, providing personalized guidance to enhance technique and understand scoring. Both initiatives use innovative tools to optimize skating performance for elite athletes and recreational enthusiasts alike. Hall, Craig R. and Wendy M. Rodgers' study examines the effectiveness of a mental skills training program for figure skating coaches. In alignment, our skating app empowers individual skaters with mental training exercises, enhancing focus, confidence, and visualization techniques. Both initiatives aim to improve skating performance by addressing crucial mental aspects for success in the sport.

Our solution to this issue is a skating companion app. The app aims to break down barriers to entry in figure skating by providing an accessible and engaging platform for learning. Recognizing that professional training can be financially prohibitive for many, the app serves as a valuable resource, offering lessons, tips, and insights into the intricate world of figure skating. By incorporating interactive elements and instructional videos, it caters to various learning styles, making the information not only informative but also entertaining. While an app is certainly no replacement for professional training, it can still assist in the learning process. The app is intended to give less experienced skaters a good head start on learning the proper rules and techniques used in real competitions, as well as to present information in an interactive and entertaining way by presenting lessons and tips accompanied by videos. In order to give players a taste of what it might be like to compete, there is also a gamification element, where a player may “train” virtual skills and watch their avatar compete in a simulated competition. Safety is also a serious concern. Injuries are common in figure skating, usually attributed to muscle overuse and long training hours [12]. Oftentimes during puberty, younger skaters get injured as the intense physical activity wears on their growing bodies [13]. The app addresses this by offering a risk-free environment for learning and practicing basic skills. In addressing issues of accessibility, safety, and education, the skating companion app aspires to be a catalyst for positive change within the figure skating community, fostering inclusivity and empowering individuals to embrace the beauty and challenges of this captivating sport.

In the first experiment, we wished to measure the effects of proficiency and success rate on the end score. In order to do this we tried various combinations of these two factors and conducted seven tests within each trial. Nine trials were performed in total, testing out high proficiency with low success, high success with low proficiency, and combinations in between. We discovered that success rate has a much more profound effect on the end score, as the points lost from failing a jump far outweigh the points gained from a flawless execution. In the second experiment, we hoped to analyze what effect proficiency did have on the score independent of success. We found that increasing proficiency from zero to ninety had about a four point average increase. While the increase was not monumental, it would be more than enough to swing the competition in a skater’s favor. The experiment also highlighted a possible flaw in our simulation, as it seemed to favor values in the extremes.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. How to Aim to Enhance Users' Skills

One of the challenges we can find when making such a skating app is how to aim to enhance users' skills even when they're not on the ice. We must find a way to create an engaging and effective set of training modules that are accessible anytime, anywhere. Augmented reality (AR) or virtual reality (VR) simulations that replicate skating environments might be a possible solution. However, the range of movement required for skating would be dangerous while wearing a modern headset, so this is not ideal. However, including instructional videos, interactive tutorials, and set personal challenges can further improve user engagement and skill development. By approaching the issue with such technology, the app can provide a well-rounded training experience, empowering users to improve their skating abilities beyond the skating rink.

2.2. Detecting and Assessing Jumps And Spins

Another challenge in the skating app lies in accurately detecting and assessing jumps and spins to provide valuable feedback. Achieving this may require a blend of sensor technologies, and machine learning algorithms. The device should be wearable, such as the one proposed by Panfili and associates The user’s device sensors should collect and label specific movements. An alternative approach would be to analyze footage of the performance, an expansion on the highlight system created by Fan and associates [5][6]. Machine learning can then step in to differentiate the different kinds of jumps and spins, improving the app's accuracy over time with user input. This process ensures there’s a precise identification of maneuvers, enabling our app to offer targeted guidance on technique and scoring strategies so judges can provide better scores. Successfully completing this challenge should enhance the app's efficacy to personalize skating skill development.

2.3. Making Sure the Backend

In creating the skating app, a major challenge posed is making sure the backend (how the app understands your moves) communicates smoothly with the outside (users actual skating). This can be tackled by using Firebase, a Database used as a smart link connecting the app on users devices and the service itself (created by Google). This communication channel between your phone and our system may allow users to exchange information about personal skating activity. However, ensuring smooth communication isn't easy – we must ensure compatibility and seamless integration so that skating movements synchronize perfectly with the app. Successfully addressing this obstacle ensures that they receive precise, instantaneous feedback on performance.

3. SOLUTION

Our simulated skating competition is managed by three distinct systems: The trainer, the planner, and the performer. The trainer manages the creation of the elements and keeps track of each move’s success rate and proficiency levels. Elements are subclassed so that different variations of moves can be interfaced the same way. The trainer has access to all the elements so that it may increase the levels upon training, and decrease them when they are neglected. The planner is in charge of creating a routine. The UI allows the user to pick a number of moves from the list and place them into their routine. Finally, the routine is passed to the performer. This system accesses the element list in the routine, and sends the corresponding data to the animator to be displayed in the scene. The element is also given a base rating calculated using the success rate and proficiency values. This rating is then sent to each individual simulated judge, each of which uses it to generate a slightly fluctuating GOE (grade of execution) value based on it. The judges then return this GOE value to the routine, which records it. Once all moves have been performed and

the their GOEs calculated, the routine is passed to the final score calculator, which reads each move's base value, each judge's provided GOE for said move, and calculates the final score using the same formula used in skating competitions as set out by the International Skating Union. These scores are finally displayed in a table for the user.

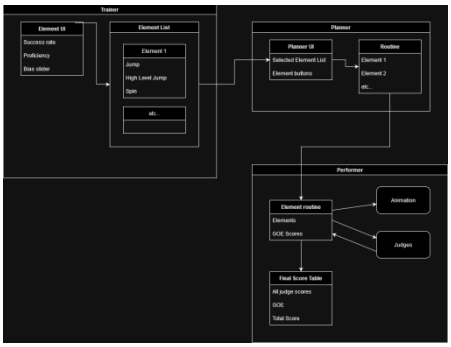

Figure 1. Overview of the solution

This system allows the user to select skills to train based on the staker's success rate and proficiency of each specific move (called elements). They are allowed to choose their emphasis of training, deciding whether they would like to focus more on consistency and execution. Their choices will influence the success rate and proficiency. The skill being trained will improve, but other skills will fluctuate up and down as they are not trained.

Figure 2. Screenshot of the app 1

```
namespace Elements
{
    public class Element
    {
        public string elementName;
        public string shortName;
        public float baseValue;
        public float successRate;
        public float goodPerformanceRate;
        public int timeTrained;
        public GameObject animationObject;

        public virtual void Train()
        {
            float improvementValue = Random.Range(1f, 6f);
            float successValue = improvementValue * (1 - GameManager.main.emphasisSlider.value);
            float goodPerformanceValue = improvementValue * GameManager.main.emphasisSlider.value;
            successRate += successValue;
            goodPerformanceRate += goodPerformanceValue;
            float fluctuationValue = Random.Range(-1.5f, 0f);
            successValue = fluctuationValue * GameManager.main.emphasisSlider.value;
            goodPerformanceValue = fluctuationValue * (1 - GameManager.main.emphasisSlider.value);
            successRate += successValue;
public class Jump : Element
{
    public Jump(string newName, string newShortName, float newBaseValue,
    {
        elementName = newName;
        shortName = newShortName;
        baseValue = newBaseValue;
        successRate = newSuccessRate;
        timeTrained = 0;
        TPC = newTPC;
        animationObject = newAnimationObject;
        goodPerformanceRate = Random.Range(30f, 60f);
    }
}
```

```
public void TrainElements(int trainIndex)
{
    for (int i = 0; i < elementList.Count; ++i)
    {
        if (i == trainIndex)
        {
            elementList[i].Train();
        }
        else
        {
            elementList[i].Fluctuate();
        }
    }
    hoursLeft -= 1;
    if (hoursLeft <= 0)
    {
        foreach (TrainPanelController TPC in trainPanels)
        {
            TPC.trainButton.interactable = false;
        }
        nextButton.gameObject.SetActive(true);
    }
    UpdateAllText();
}
```

Figure 3. Screenshot of code 1

The code segment shows the element class, which is used to represent a move. Jumps and spins are subclasses of the element, allowing them to have their own properties while still interfacing with our train/fluctuate methods (high-level jump derives from jump). This structures data in a logical way, minimizes repetition of code, and allows modularity. The code snippets above show the class declarations and their constructors. In order to allow emphasis on either consistency or skill, the train method reads the value on the bias slider. It picks random values between 1-6 and splits the resulting points between the two attributes.

The Train Elements method acts as a controller here, increasing the attributes of the selected element (each element has a train method which is overridden on different children to yield different effects / change diff values, which means jumps train different than spins) when training, while simultaneously fluctuating each other element to simulate how not practicing causes a general decrease in skill.

This system allows the user to plan out their routine. They are presented with seven empty slots, and a selection of buttons down below. Clicking a button will add it to the selected slot, indexing the selector by one. The user may click on a specific element if they wish to replace that one. To the right of each element is its base score, which is totaled at the bottom.

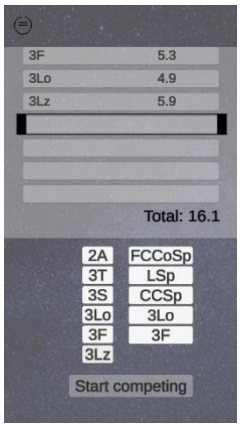

Figure 4. Screenshot of app 2

```
public void AddElementsToList(int index)
{
    //shortRoutine.AddElement(jumps[index]);
    curRoutine.OverwriteElement(elements[index], selectedIndex);
    print(elements[index].elementName);
    if (selectedIndex != curRoutine.elementList.Length - 1)
    {
        selectedIndex += 1;
        SelectPanel(selectedIndex);
        panelArray[selectedIndex].SelectMe();
    }
    UpdateAllPanels();
}
public void UpdateAllPanels()
{
    int routineLength = curRoutine.elementList.Length;
    for(int i = 0; i < routineLength; ++i)
    {
        print(i);
        PerformanceElement currentElement = curRoutine.elementList[i];
        string currentElementName = "";
        string currentElementBaseValue = "";
        if (currentElement != null)
        {
            currentElementName = currentElement.getShortName();
            currentElementBaseValue = currentElement.getBaseValue().ToString();

        }
        panelArray[i].UpdateText(currentElementName, currentElementBaseValue);
    }
    totalText.text = "Total: " + curRoutine.GetTotalBaseValue();
    startButton.interactable = curRoutine.isComplete();

}
```

```
public class Routine
{
    public PerformanceElement[] elementList;
    int maxElements;

    public bool[] successArray;
    public bool[] goodLandingArray;

    public Routine(int newMax)
    {
        maxElements = newMax;
        elementList = new PerformanceElement[maxElements];
        successArray = new bool[maxElements];
        Debug.Log("hello" + elementList.Length);

    }

    public float GetTotalBaseValue()
    {
        float total = 0;
        foreach (PerformanceElement element in elementList)
        {
            if (element != null)
            {
                total += element.getBaseValue();
            }
```

Figure 5. Screenshot of code 2

The Routine class stores all information relating to a routine, such as the elements the player plans to perform in the routine and whether each element has been performed successfully. It also contains methods to get the total base value and overwrite or add elements, which are useful when displaying to the UI and also when calculating the final score after the competition. When adding an element to a list, the AddElementToList method is called, which overwrites a particular element in the routine before indexing the selector to the next empty space. The UpdateAllPanels method controls the UI. When it is called, it fetches the routine instance and extracts the element name and base value. It then accesses each corresponding panel and assigns this information. It also controls the start button, not allowing the user to press it until they have filled all the slots.

This component simulates skating competitions based on the routine they've planned and the training choices the player has made. The animation corresponds to the simulated results, showing the player performing moves and the judges giving their individual scores. Once the competition is over, the player is shown a table showing their performance in the competition. It displays the base value, grade of execution, each judge's score, and the total points.

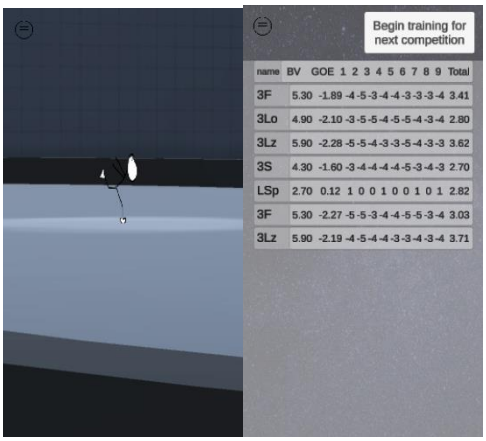

Figure 6. Screenshot of app 3

```
public IEnumerator CompetitionCoroutine()
{
    aud.Play();
    mainCamera.SetActive(false);
    for (int i = 0; i < currentRoutine.elementList.Length; i++)
    {
        foreach (JudgeScript judge in judgeArray)
        {
            judge.ResetCard();
        }
        judgeCamera.SetActive(false);
        GameObject instance = Instantiate(currentRoutine.elementList[i].getAnimationObject(), skater
        SelectRandomCam(instance.transform.GetChild(0));
        AnimatorClipInfo[] clip = instance.GetComponent<Animator>().GetCurrentAnimatorClipInfo(0);
        float skaterMoveDelay = clip[0].clip.length;
        yield return new WaitForSeconds(skaterMoveDelay);
        currentRoutine.elementList[i].perform();
        judgeCamera.SetActive(true);
        DisableAllCams();
        print(currentRoutine.successArray[i]);
        foreach (JudgeScript judge in judgeArray)
        {
            judge.GiveScore(currentRoutine.elementList[i], currentRoutine.successArray[i]);
            yield return new WaitForSeconds(judgeDelayTime);
        }
        yield return new WaitForSeconds(judgeDelayTime * 9 + 0.5f);
        Destroy(instance);
    }
    judgeCamera.SetActive(false);
    mainCamera.SetActive(true);
    aud.Stop();
    onPerformanceFinish.Invoke();
}

 public int GiveScore(PerformanceElement elementPerformed, bool succeeded)
 {
     int score = 0;
     if (elementPerformed.succeeded)
     {
         score = Random.Range(0, 2);
         if (elementPerformed.goodPerformance)
         {
             score = Random.Range(2, 4);
         }
     }
     else
     {
         score = Random.Range(-3, -6);
     }
     score = Mathf.Clamp(score, -5, 5);
     DisplayScore(score);
     addScoreToList(score);
     return score;
 }
```

Figure 7. Screenshot of code 3

The Competition Coroutine is a coroutine defining the flow of the simulated competition. It first begins by playing the music, then disables the main camera. It then enters a for loop which iterates through each element in the list. There is a Judge class which simulates the individual judges present at a competition. The code first begins by resetting all of the judges, then instantiating the animation corresponding to the element that should be performed and calculating whether or not it should be a success. It then picks a random camera within the rink to enable for the animation. After waiting for the animation to finish, it enables the judge camera, which shows the judge panel. Each judge contains a GiveScore method, as seen above. This method gives a fluctuating score based on whether the move succeeded, and also whether it was considered a good performance. After the judges are finished giving their scores, the loop cycles again and the next element is performed. This is repeated until all elements in the routine have been performed.

4. EXPERIMENT

4.1. Experiment 1

When we designed the judges, we wished to make the scoring system reflect reality as closely as possible. Thus, we used the same formulas and calculations that are used in real competitions. However, since the score was passed through so many formulas, it made it difficult to determine how much our statistics (for proficiency and success) were affecting the score.

To test this, we tried varying levels of proficiency and success rate, using only one of the base moves in order to reduce confounding variables. In this case we chose the double axel. We then created a routine using only this move, and analyzed the end score for each performance. We tried both low proficiency with high success, high proficiency with low success, and high and low of both. Since there are seven moves in each routine, each set of proficiency and success rates had seven trials. This is done for ten tests in total.

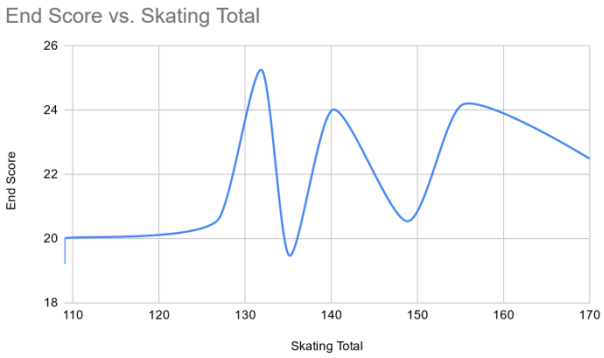

Figure 8. End Score vs. Skating Total

Figure 9. Proficiency, Success Rate, and End Score

In the first graph, success rate and proficiency are combined into a "skating total" to see the effect of both variables together. As we can see there is a general upwards trend, which is to be expected as higher levels of both skills should lead to a higher overall score. However, there is a significant amount of fluctuation in the data, which makes it difficult to make out the trend definitely. One thing to note is that the increase in score was not nearly as dramatic as we thought it would be, though this could be due to the fluctuation making it difficult to discern the trend.

In the second graph, we looked at each individual move and sorted them based on the score they got. We then graphed the proficiency and success rate as two different points in a scatter plot to determine which has a more profound effect on the score. As we can see, on the higher end of the spectrum there are more red dots above the blue dots. This indicates that, though both stats have an effect on the end score, success rate is the more important factor. This makes sense, as even if a skater can perform a very complicated move that earns a lot of points, it will do them no good if they lose those points by failing the jump.

4.2. Experiment 2

We know that success rate is the most important factor when it comes to determining score. However, though being able to avoid a negative penalty is good, it is also important to actually perform the move well, and we want to ensure our simulation reflects that. This experiment was designed to determine the effect of proficiency on the end score independent of success or failure.

In order to test this, we performed ten separate trials. Each trial had a fixed proficiency level. Trial one had a proficiency level of zero, trial two had ten, trial three had twenty, and so on. Trials stopped once ninety proficiency was reached. Seven tests were performed for each trial, after which the score for each move and the total score for the routine were recorded. Similar to the last experiment, the double axel was chosen as the basis for this experiment. For all trials, the success rate was fixed at 100% to ensure that the results were a product of the proficiency level only.

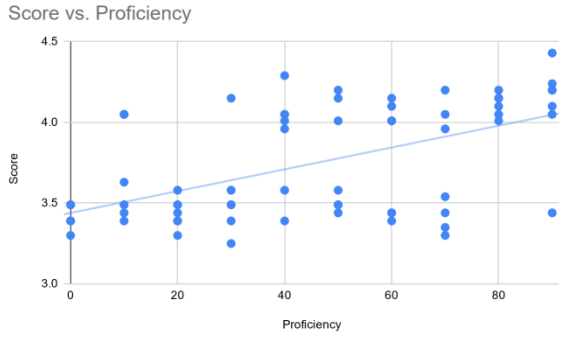

Figure 10. Score vs. Proficiency

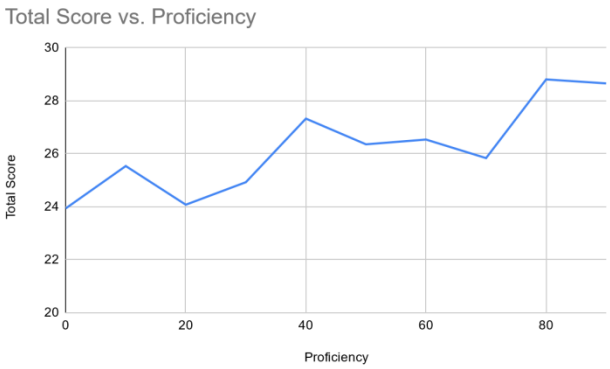

Figure 11. Total Score vs. Proficiency

The top graph documents the score of each individual move, while the bottom one documents the total score of the routine. As is evident in both of these graphs, there is a clear upward trend as proficiency increases. There is an average increase of around 4 points between the total scores of simulations with no proficiency vs ninety. While this increase may seem slim, every point counts in the high-level competitions that this simulation is intended to mimic. One thing to note is that there is a visible gap in the top graph between the high and low scores. This seems to highlight a quirk in the way our program calculates score, favoring either low-end scores or high-end ones. This could be because our program internally has a threshold that is considered a "good" performance of the move, and when a move falls above/below this threshold our simulated judges tend to gravitate more heavily towards the extremes.

5. RELATED WORK

The study by EJ Kovacs, TB Birmingham, L Forwell, et al. aims to investigate the impact of two distinct training programs on the postural control of figure skaters. One program focuses on neuromuscular training, targeting muscle strength and coordination, while the other focuses on a more conventional off-ice training program. Through a randomized controlled trial, researchers found which approach provides better outcomes in enhancing skaters' ability to maintain balance and control during practice [4]. By rigorously comparing these interventions, the study focuses to provide solid evidence on the most effective training strategies for the best postural control among figure skaters, which would inform future training protocols and performance enhancement strategies better. They concluded off-ice neuromuscular training improves figure skaters postural control.

The research by Eb, Jeroen van der, Sjoerd Gereats, and Arno Knobbe introduces SkateView, a device designed to boost the performance of elite speed skaters by providing precise measurements during training and competitions [1]. Similarly, our skating app aims to make use of applications as a technology to enhance users' skating skills and performance. While SkateView focuses on elite athletes, our app caters to all skating enthusiasts, offering personalized feedback and guidance to help users improve their technique and understand how to score points with judges. Both initiatives leverage innovative tools to optimize skating performance, whether at the elite level or for recreational skaters seeking to enhance their skills and confidence on the ice.

The study by Hall, Craig R., and Wendy M. Rodgers explores how a mental skills training program can make coaching in figure skating more effective [10]. Similarly, our skating app enhances users' skills by offering mental training exercises and strategies. While the study focuses on coaches, our app empowers individual skaters, providing guidance on mental aspects like focus, confidence, and visualization techniques. By integrating mental skills training into our app, users can develop not only physical skills but also the mindset needed to perform their best on the ice and impress judges. Both initiatives aim to improve skating performance by addressing mental aspects crucial for success in the sport.

6. CONCLUSIONS

There were a number of limitations in our project due to time constraints, and we were not able to complete all of the goals we had originally sought out to do. The simulation we have created which allows the user to make choices and see a virtual skating competition gives a better idea of the rules and how a competition would play out, but there is very little interactivity. It would be ideal if the player could, for example, swipe up to jump during the performance. This would improve interactivity and maintain player attention. Additionally, we would also like to simulate

other skaters to give the game more of a competitive feel. Finally, while we did create code infrastructure to transport data over Firebase, as well as created a basic hardware prototype, we were unable to integrate them all together as a single unit at the time of this paper. Perhaps this can be done at a later date.

Figure skating is a difficult sport to get into, not only because of the steep learning curve and costs associated with it, but also the nuances a skater must navigate if they wish to please the judges. This application aims to reduce the learning curve slightly by providing an accessible and simple interface to facilitate training. While the app cannot replace professional training, it can certainly assist those who have no other options.

REFERENCES

[1] Eb, Jeroen van der, Sjoerd Gereats, and Arno Knobbe. "Enhancing the Performance of Elite Speed Skaters Using SkateView: A New Device to Measure Performance in Speed Skating." Proceedings. Vol. 49. No. 1. MDPI, 2020.
[2] Albert, Wayne J., and Doris I. Miller. "Takeoff characteristics of single and double axel figure skating jumps." Journal of Applied Biomechanics 12.1 (1996): 72-87.
[3] Jederström, Moa, et al. "A cross-sectional study of anxiety and depression caseness in female competitive figure skaters in Sweden." BMJ Open Sport—Exercise Medicine 9.1 (2023).
[4] Kovacs, Emily J., et al. "Effect of training on postural control in figure skaters: a randomized controlled trial of neuromuscular versus basic off-ice training programs." Clinical journal of sport medicine 14.4 (2004): 215-224.
[5] Panfili, Antonio, Alvise Spanò, and Agostino Cortesi. "A Wearable System for Jump Detection in Inline Figure Skating." Sensors 22.4 (2022): 1650.
[6] Fan, Shun, et al. "Hightlight Video Detection in Figure Skating." Chinese Conference on Pattern Recognition and Computer Vision (PRCV). Cham: Springer Nature Switzerland, 2022.
[7] Findlay, Leanne C., and Diane M. Ste-Marie. "A reputation bias in figure skating judging." Journal of Sport and Exercise Psychology 26.1 (2004): 154-166.
[8] Seltzer, Richard, and Wayne Glass. "International politics and judging in Olympic skating events: 1968-1988." Journal of Sport Behavior 14.3 (1991): 189.
[9] Stepanova, Elena V., Michael J. Strube, and John J. Hetts. "They Saw a Triple Lutz: Bias and Its Perception in American and Russian Newspaper Coverage of the 2002 Olympic Figure Skating Scandal 1." Journal of Applied Social Psychology 39.8 (2009): 1763-1784.
[10] Hall, Craig R., and Wendy M. Rodgers. "Enhancing coaching effectiveness in figure skating through a mental skills training program." The Sport Psychologist 3.2 (1989): 142-154.
[11] Cummins, Lori F. "Figure skating: A different kind of youth sport." Journal of Clinical Sport Psychology 1.4 (2007): 390-401.
[12] Han, Julie S., Ellen T. Geminiani, and Lyle J. Micheli. "Epidemiology of figure skating injuries: a review of the literature." Sports health 10.6 (2018): 532-537.
[13] Dubravcic-Simunjak, Sanda, et al. "The incidence of injuries in elite junior figure skaters." The American journal of sports medicine 31.4 (2003): 511-517.
[14] Lee, Jungmin. "Extreme Sports Evaluation: Evidence from Judging Figure Skating." Econometric Society 2004 North American Summer Meetings. No. 122. Econometric Society, 2004.
[15] Grenfell, Christopher C., and Robert E. Rinehart. "Skating on thin ice: Human rights in youth figure skating." International Review for the Sociology of Sport 38.1 (2003): 79-97.