Kalman-filter for partial linear systems for tracking pool balls

Marlon Lückert
Bachelor of Science
marlon.lueckert@haw-hamburg.de

Julius Neudecker
Bachelor of Science
julius.neudecker@haw-hamburg.de

June 2020

Contents

In this document we are going to discuss an advanced implementation of the Kalman-filter [1] improve measurement quality and predict the movement of balls on a pool table. We are going to point out the necessity of an advanced filter design in this particular case. The Problem we are going to solve is the following: Since the performance of the filter deteriorates in cases of a rapid change in direction, the filter has to be able to adapt more quickly to these rapid changes. Our plan is to derive an implementations with adaptive behavior. The implementation will be tested in a simulator and with real world video footage of a pool table. The goal is to provide a filter design with a significant lower MSE than a vanilla implementation. Furthermore we will point out our methods of research, literature sources and provide a deep insight in our data analyzation.

1 Introduction

At first glance the game of pool is very suitable to examine the behavior of a kalman-filter enhanced tracking system based on pure visual tracking. The surface of a pool table is made of a thin fabric which covers a hard surface i.e. slate or granite. The balls nowadays are usually made out of resin. This combination of materials creates very small rolling resistance and the balls behave almost fully elastic on collision. Since this is only a 2 DOF[1] problem, this can be solved with a simple linear kalman filter implementation. The problem is, when two balls hit each other or a cushion the velocity vector changes its orientation instantly. If this isn't taken into account, the filter needs some time to adapt to the new direction of movement and will produce wrong estimations during this time.

This behavior is independent of the type of kalman implementation being constant-velocity-model or constant-acceleration-model. A kalman filter will assume the direction of movement on any given sample is about the same as in the last sample. It will therefore create wrong estimations if the direction of movement changes drastically in a short period of time. The time the filter needs to recover depends on the filter gain.

We also use the filter to predict values for any given length into the future by feeding back its estimations as actual state. The quality of this prediction however depends on several factors, i.e. the applied process noise and framerate of the video.

2 Problem and Motivation

As introduced in the previous section is the lack of adaptability in a vanilla Kalman implementation. One could argue that this can be taken into account by using a higher overall process noise which would in turn lead to significantly reduced overall quality. This gets worse with increased speed and lower framerate. Essentially rendering the filter useless at a certain point.

We think this is a good point to develop this algorithm for future application in VR and AR based pool trainers and augmented broadcast experiences.

3 Hypothesis and Goals

We can improve the algorithm in the Kalman-Filter by taking the environment into account. If we feed the algorithm with the position of objects which could cause a collision, the filter can dynamically react to those sudden changes in the direction of movement.

4 State of research

Jong-Yun Kim and Tae-Yong Kim [2] developed a method to provide robust tracking of a soccer ball. They provide a solution for the problem for the case that the soccer ball might be occluded by the player at any given time, which results in a diminished tracking accuracy. In this case they used the velocity vector of the player to substitute for the ball presuming that the ball moves in the same direction as the player does.

Jia et.al. [3] conducted research in the trajectory of pool balls, which helped us to decide which kalman model is the most suitable.

Shiuh et.al. [4] provided a good starting point how to create a tracking algorithm for pool balls. They also developed an algorithm to track occluded objects using an adaptive kalman filter. In this case they used to threshold in order to determine whether the object can still be reliably tracked. If this isn't the case the filter will rely only on predicted values until the object can be tracked reliably again.

[1] Dimensions Of Freedom - determines the possible rotation or translation along each given axis

Salzmann and Urtasun [5] proposed a more general approach for tracking. They were able to recreate a highly accurate tracking from a noisy picture based on newtons 2nd law and markov models. Using different constraints and presumptions they were even able to extract physical parameters like friction and trajectories.

Mohamed and Schwarz [6] are using partly the same approach as we do to improve the results created by INS/GPS[2] Systems. However their approach only targets the 'Q' and 'R' parameters of the filter.

Sarkka and Nummenmaa [7] created an adaptive kalman implementation which adapts itself to time-varying noise parameters. Since our input data is constant in this regard, we decided to simulate for the optimal filter parametrization instead of relying on the filter to adapt itself.

Gabdulkhakova and Kropatsch [8] use a kalman filter to create a video analysis tool for snooker game broadcasting.

5 Methods of research

We are going to use two stepped approach: develop and the algorithm in a tailor made simulation and evaluate the results with real footage video. Our main instrument of validation will be the MSE [3] between the ground truth and filtered position and the quality of prediction with +n frames respectively.

We are going to use a two-stepped approach:

Firstly we will optimize the process noise behaviour in edge cases. Meaning that at points where the ball is at risk of a sudden change in direction, the filter adapts itself by temporarily altering its process noise parameter. Thus being less prone to overshooting the critical point. We are planning to use a gradient descent algorithm in order to optimize the filter behavior in all states.

Secondly since a rapid change in direction is not a linear problem in terms of implementation of a kalman filter, we are plannning on using linear algebra to reduce this problem to a linear vector multiplication. By doing so we will achieve not only a sudden but also precise change without utilizing the time invariant input capabilities of the filter.

6 Research Goals

Primarily we are looking for related work in the field of kalman filtering in billiards, to inform ourselves about the current progress of research and state-of-the-art solutions.

Secondly our goal with the literature research is to find sources of information which complements our expertise.

While writing our proposal we identified five aspects where we lack sufficient expertise. These are in particular:

Design of experiments As outlined in the research proposal we are going to develop our ideas in a simulation. Therefore it is efficient to design the experimental part in advance. This means that we are not going to use a trial-and-error approach but instead invest some time to think about possible outcomes and how to achieve these by altering different parameters in particular and develop a deeper understanding of their behavior.

Gradient descent algorithms Since there exist a huge number of different numerical and analytical ways to solve a gradient descent problem it will save time to evaluate the best suitable algorithm prior to implementation to avoid unnecessary work.

[2] Inertial Navigation System / Global Positioning System
[3] Mean Square Error

Efficient implementation of algorithms This is especially important since our algorithm is supposed to work in realtime. Therefore any delay induced by inefficient code is unacceptable.

openCV best practices The analysis of the original footage is made by openCV. Since this library offers a wide variety of different approaches to isolate objecs in videofootage, it is evaluate the best option.

Measure and comparison of quantifiable data The results in our research will be mainly represented by numbers. To create a better understanding of these numbers and how they interact it is a good approach to evaluate different methods of putting numbers in context to one another and which scales and graphs are the quasi standard across the research community.

7 Source of Information

For the research regarding *state-of-the-art solutions* and *efficient implementation of algorithms and openCV best practices* it is the best option to look for recent papers since these topics are currently subjects where a certain number of people are working on. Therefore the following archives offer a good starting point to search for up-to-date papers:

1. IEEE online database
2. arXiv.org
3. ACM Digital Library
4. Google Scholar

The Topics *statistical experimental design and measure and comparison of quantifiable data* are already established in the scientific community. Therefore it is the best approach to consult the latest literature in these particular fields. In our case this can be done to a certain extent with the HIBS database and online services or with actual books in the library.

The efficient implementation of algorithms is less of a scientific problem but of a code-optimization. Hence reading blogs of skilled programmers or threads on *Stackoverflow* seems as the most reasonable approach.

8 Criteria for eligible sources

The general criteria for any sources are mostly common sense:

- credibility and plausibility
- a certain standard of quality
- integrity

These criteria might seem overly cautious in case the source is a book written by any member of the scientific community. However in the case of short paper or online blog the author of that particular source might not be as credible. Therefore a higher level of scrutiny with the informations provided is necessary.

Furthermore since the sources might be cited in a paper or dissertation they have to be scientificly quotable. This is expecially true to protect the resulting work against accusations of plagiarism.

As already pointed out in the previous section some topics are more subject to change due to present research activities and must therefore reflect the most recent state of research.

Another important aspect is especially the case with non-scientific online ressources as blogs and the previously mentioned *Stackoverflow*. Since these sources are also subject to change, it is important to keep track of the particular version of the ressource to prevent confusion in case the ressources changes.

9 Sources for our research

9.1 Related Work

Jong-Yun Kim and Tae-Yong Kim [2] developed a method to provide robust tracking of a soccer ball. They provide a solution for the problem for the case that the soccer ball might be occluded by the player at any given time, which results in a diminished tracking accuracy. In this case they used the velocity vector of the player to substitute for the ball presuming that the ball moves in the same direction as the player does.

Jia et.al. [3] conducted research in the trajectory of pool balls, which helped us to decide which kalman model is the most suitable.

Shiuh et.al. [4] provided a good starting point how to create a tracking algorithm for pool balls. They also developed an algorithm to track occluded objects using an adaptive kalman filter. In this case they used to threshold in order to determine whether the object can still be reliably tracked. If this isn't the case the filter will rely only on predicted values until the object can be tracked reliably again.

Salzmann and Urtasun [5] proposed a more general approach for tracking. They were able to recreate a highly accurate tracking from a noisy picture based on newtons 2nd law and markov models. Using different constraints and presumptions they were even able to extract physical parameters like friction and trajectories.

Mohamed and Schwarz [6] are using partly the same approach as we do to improve the results created by INS/GPS[4] Systems. However their approach only targets the 'Q' and 'R' parameters of the filter.

Sarkka and Nummenmaa [7] created an adaptive kalman implementation which adapts itself to time-varying noise parameters. Since our input data is constant in this regard, we decided to simulate for the optimal filter parametrization instead of relying on the filter to adapt itself.

Gabdulkhakova and Kropatsch [8] use a kalman filter to create a video analysis tool for snooker game broadcasting.

In the article [9] they use a Kalman filter for smooth estimations of a billiards game.

In [10] they present an adaptive kalman filter, which estimates Q and R values.

9.2 Design of experiments

To get a general sense of how statistical experimental design works these Books provide a good introduction into this topic: [11] and [12]. Both books are available at our local Library.

A more practical approach is outlined in the articles [13] and [14].

[15] is a good source of examples to put everything in context.

9.3 Gradient descent algorithms

Although the gradient descent method is often used in Machine Learning to optimize parameters in neural networks, we can use it to solve our optimization problem. The

[4]Inertial Navigation System / Global Positioning System

articles [16] and [17] provides an introduction to the optimization problem with GD methods whereas [17] gives an overview with more advanced methods.

[18] is a very famous book for more advanced topics. Since we've got a basic understanding of how to implement GD in our simulation, we might consider using some advanced techniques from this book. This might prove especially useful because the normal gradient descent method can only find local minimas.

9.4 Efficient implementation of algorithms

There exist a few different approaches to optimize an algorithm for fast execution. These are in particular paralelization and compiletime optimization.

Since our optimized filter will highly rely on vector and matrix operations, there is a big potencial for all these optimizations. However since we lack experience in writing special code like this, we have to make us familiar with the basics of such coding.

For paralelization through parallel hardware we could use openCL. An comprehensive introduction is provided by [19] and [20]. Depending on the hardware we also might go with a proprietary GPGPU framework such as CUDA. In case we should decide to use this [21] is an introduction into this topic.

An introduction in compiletime optimization is provided by [22] and introduction on how to use the new AVX instruction on Intel Processors is given by [23].

9.5 openCV best practices

With the article [24] we can get an overview and comparison of different detection algorithms for the OpenCV framework.

The article [25] gives an insight on how to design a multi-object detection system for billiards in openCV, which perfectly fits our research case.

The conference paper [26] uses neural networks in conjunction with openCV to detect balls. This approach aims for an efficient re-training of neural networks. This is useful for our research, because we can easily adopt the algorithm to different pool environments and do not have to manually adjust the ball detection.

The thesis [27] focuses on performance optimization for speed estimation of balls with openCV on mobile devices. This is useful for our research, because we want to achieve real-time tracking and eventually port the application to mobile devices.

9.6 Measure and comparison of quantifiable data

Because error analysis is the driving factor behind our research since we want to minimize the error which is produced by an inferior filter-design, we have to make ourselves familiar with state of the art error analysis and the uncertainties of measurement with [28].

[29] provides a good explanation why in predictions it is better to use the MSE instead of the LSE.

The standard method to visualize the mean square error of Kalman-filters are 2D plots, which can be seen in [30] and [31]. The y-axis describes the MSE and x-axis the observed parameter.

10 Proposed timetable

To conduct our research in a timely manner, we propose the following schedule to have the finished research paper by the end on June 2020.

- until End of June: develop the mathematical framework
- until End of July: develop the simulation and create footage for evaluation

- until Middle of August: implement the filter according to optimized parameters
- until Middle of September: conclude results and write research paper

We are planning to submit this paper to a conference which has relevance in this particular field of research. This has yet to be determined.

11 Operationalization

In the following section we are going to operationalize the different terms in our hypthesis.

11.1 Tracking

The tracking describes the deviation from the filtered position of the billiard ball (result of Kalman-Filter) to the real position of the billiard ball. The position divides into three dimensions, the x-coordinate, the y-coordinate, the z-coordinate. Since we are investigating a billiard game we disregard the up-direction (z-coordinate), because the ball moves on a flat surface and will not leave the plane. We are measuring the coordinates in pixels. The range of the pixels is defined by the resolution of the input video on which the Kalman-Filter gets applied to. The origin, coordinate (0,0), is always the top-left corner. The final deviation from the two position is calculated with the Euclidean distance, which results in a length in pixels.

The position of the billiard ball changes over time, because we are investigating video sources. To measure the deviation of the whole video and to get a key figure, we are calculating the mean square error (mse) in which the error is the pixel distance between the two positions.

11.2 Prediction

The prediction describes the deviation from the calculated future position of the billiard ball to the real position at this specific time. The positions use the same measurements as described in the previous chapter. The time is measured in frames. The specific number of frames we will look in the future will vary in our experiments. To make the frame count comparable with different input video source we align them with the framerate (frames per second) of the video.

11.3 Improvement

The improvement compares a standard Kalman-Filter to our implementation of the Kalman-Filter. The improvement is measured as the difference between the different mean square errors of the tracking and the prediction. The smaller the mse of our Kalman-Filter compared to the standard implementation the greater the improvement.

12 Data Acquisition

We are going to acquire our data from billiard simulations and real videos.

12.1 Simulation

We construct a simulation to fully control every parameter and the state of the billiard game. This helps us to accomplish a better conclusion to the performance in real video. Our examination units are the framerate, the ball velocity, the sensor noise.

Framerate The framerate describes the framerate of the video, which is equal to the sampling rate of the Kalman-Filter. A standard Kalman-Filter performs much worse on low framerate because it gets less data to process. We test different framerate to inspect the performance of our implementation for different environments. For realtime applications on mobile devices our implementation has to perform well on low framerates. We are going to test the standard framerates 30 and 60 FPS and a very low framerate at 10 FPS.

Ball Velocity The ball velocity describes the speed of the ball, which depends on how hard the player hits the ball. The velocity is measured in pixel per frame. The cover a lot of cases, we choose a low velocity of 300 pixel per frame, an average velocity of 500 pixel per frame and a high velocity of 700 pixel per frame. We have to keep in mind that a high velocity also leads to more collisions because the ball will move a longer distance.

Sensor Noise When we analyze realtime videos we have detect the balls in the video frame. The detection is not always perfect and heavily relies on the light conditions and resolution of the camera. To simulate the quality of detection we added a noise value which changes the detected ball position by random value in a certain range. We test a deviation of 15 pixel, 30 pixel and 60 pixel to inspect how well the Kalman-Filter will perform.

Change of parameters To test the parameters individually we only change one parameter at the time. Three variations on each parameter lead to 9 simulation environments in total.

12.2 Real Videos

After testing the filter in the simulation we are also doing some test on random billiard videos. Since we are relying on a ball detection algorithm to identify the billiard balls, we cannot obtain the real position of the ball like we could in the simulation. That's why we cannot make statements about the tracking quality, but we can evaluate the performance of the tracking.

13 Data Evaluation

In order to evaluate if and how much better our filter implementation performs compared to the standardized Kalman-Filter we have to run tests and find a way to objectively compare the results. In the next sections we are going to discuss the test environment and the data we acquire from it.

The key figure of evaluation is the mean square error. If the means square error of our implementation is lower compared to the standard implementation in every experiment we can prove that our implementation improved the tracking and prediction of billiard balls.

14 Testing

Our Kalman-Filter is specifically designed to track and predict the position of billiard balls. The final goal is to implement this filter in a mobile app, which analyzes video footage preferably in realtime.

To test the filter we compare calculated position of the filter to the real position of the billiard pool.

But we cannot test our filter with video material because we do not know the ground truth of the balls' position in the video. So we are not able to make statements about the quality of the filter if we cannot compare the filter result to the optimal result.

Therefore we have to develop a virtual test environment where we can control every parameter of the billiard game and know the actual values of the balls' position.

14.1 Simulation

The simulation has to reflect all characteristics and physical conditions of a real billiard table. The virtual environment consists of a billiard ball and four cushions. We can control the start acceleration of the ball where we can reflect start shots with different intensity. When the ball starts moving a constant deceleration is applied to mimic the friction of the table's fabric. If the ball touches a cushion the direction of movement is mirrored to reflect a collision.

Our simulation does not include the rotation of the ball which would leads to a different behavior on cushion collisions. We have to assume that the ball is hit directly in the middle with no rotation.

With the simulation we can access all parameter of the ball like the position, speed and acceleration.

14.2 Modifying the simulation

The purpose of a Kalman-filter is to improve noised sensor data and filter bad measurements. When we analyze video footage the ball detection algorithm works as a sensor which provides the estimated position of the ball. The quality of this detection depends on different factors like the video resolution or the contrast of the images. But the sensor result will never be exactly the ground truth.

With the simulation we have perfectly correct values. So we have to modify the correct values by adding noise to reflect the real world circumstances. We have full control over the amount of noise, which makes it possible to simulate different qualities of sensors.

The simulation also provides data every time we want them. But in a real example you are limited to the framerate of the video and the performance of the ball detection algorithm. Therefore we have to define a sampling rate on which we process the data of the simulation.

14.3 Parameters for testing

After we defined the technical frame we can adjust different parameter to modify the simulation.

- start acceleration. How hard is the first hit of the ball?
- sensor noise. How close is the estimated position to the real position?
- sampling rate. How often is it possible to process a video frame?

We define three values for each parameter and compare them among each other which results in nine tests.

For the start acceleration we choose a slow hit with 80 cm/s, a medium hit with 1 m/s and a power shot with 4 m/s [32]

For the sensor noise we choose a deviation of 10 pixels, 50 pixels and 100 pixels.

For the sampling rate we choose 15 fps (very slow algorithm), 30 fps (realtime on typical mobile devices), 60 fps (realtime on modern devices).

Each of the nine tests are run with the standardized Kalman-filter and our filter implementation.

14.4 Data acquisition

With each test run we collect the estimated position with the normal Kalman-filter, our implementation and the ground truth. Additionally to the current position we collect the prediction of the position in 0.2 seconds, 1 second and 2 seconds.

15 Evaluation

To compare the results we calculate a single number which indicates the quality of each filter in each test. The indicator is the mean square error, which is calculated with the difference between the ground truth and the filter result in each frame of the testing simulation. After that we calculated the mean difference throughout the whole simulation. The smaller the mean square error the better performs the filter.

Since we are running multiple test with different parameters we can make statements about what parameter has the biggest influence of the quality of the filter.

References

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, 1960.

[2] J.-Y. Kim and T.-Y. Kim, "Soccer ball tracking using dynamic kalman filter with velocity control," *IEEE, Tianjin, China*, 2009.

[3] Y.-B. J. et. al., "Trajectory of a billiard ball and recovery of its initial velocities," *Department of Computer Science Iowa State University*, 2011.

[4] S.-K. et. al., "Video object tracking using adaptive kalman filter," *Journal of Visual Communication and Image Representation*, 2006.

[5] M. Salzmann and R. Urtasun, "Physically-based motion models for 3d tracking: A convex formulation," *International Conference on Computer Vision*, 2011.

[6] A. H. Mohamed and K. P. Schwarz, "Adaptive kalman filtering for ins/gps," *Journal of Geodesy*, 1999.

[7] S. Sarkka and A. Nummenmaa, "Recursive noise adaptive kalman filtering by variational bayesian approximations," *IEEE Transactions on Automatic Control*, 2009.

[8] A. Gabdulkhakova and W. G. Kropatsch, "Video analysis of a snooker footage based on a kinematic model," *Vienna University of Technology*, 2012.

[9] F. Wu and A. Dellinger, "Capturing reality for a billiards simulation," in *Augmented Reality, Virtual Reality, and Computer Graphics* (L. T. De Paolis, P. Bourdot, and A. Mongelli, eds.), (Cham), pp. 289–298, Springer International Publishing, 2017.

[10] S. Akhlaghi, N. Zhou, and Z. Huang, "Adaptive adjustment of noise covariance in kalman filter for dynamic state estimation," in *2017 IEEE Power Energy Society General Meeting*, pp. 1–5, 2017.

[11] K. Siebertz, D. van Bebber, and T. Hochkirchen, "Statistische modellbildung," in *Statistische Versuchsplanung*, pp. 87–137, Springer, 2017.

[12] G. Retzlaff, G. Rust, and J. Waibel, *Statistische Versuchsplanung: Planung naturwissenschaftlicher Experimente und ihre Auswertung mit statistischen Methoden.* Verlag Chemie, 1978.

[13] F. Hoevelmann, H. Otten, and D. Pohl, "Statistische versuchsplanung einmal anders," *QUALITAT UND ZUVERLASSIGKEIT*, vol. 38, pp. 285–285, 1993.

[14] W. Schweitzer and C. Baumgartner, "Off-line-qualitätskontrolle und statistische versuchsplanung," *Zeitschrift für Betriebswirtschaft*, no. 1, pp. 75–100, 1992.

[15] K. Siebertz, D. van Bebber, and T. Hochkirchen, "Doe beispiele," in *Statistische Versuchsplanung*, pp. 159–177, Springer, 2017.

[16] N. KETKAR, "Deep learning with python, a hands-on introduction, apress edition, 160p," 2017.

[17] K. Marti, *Stochastic optimization methods*, vol. 2. Springer, 2005.

[18] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[19] N. Trevett, "Opencl introduction," *Khronos Group*, 2013.

[20] J. Tompson and K. Schlachter, "An introduction to the opencl programming model," *Person Education*, vol. 49, p. 31, 2012.

[21] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," *Queue*, vol. 6, no. 2, pp. 40–53, 2008.

[22] M. Hohenauer, C. Schumacher, R. Leupers, G. Ascheid, H. Meyr, and H. van Someren, "Retargetable code optimization with simd instructions," in *Proceedings of the 4th international conference on Hardware/Software Codesign and System Synthesis*, pp. 148–153, 2006.

[23] M. Cornea, "Intel avx-512 instructions and their use in the implementation of math functions," *Intel Corporation*, 2015.

[24] P. Janku, K. Koplik, T. Dulik, and I. Szabo, "Comparison of tracking algorithms implemented in opencv," *MATEC Web of Conferences*, no. 76, 2016. Available at: `https://www.matec-conferences.org/articles/matecconf/pdf/2016/39/matecconf_cscc2016_04031.pdf`.

[25] J. Gao, Q. He, H. Gao, Z. Zhan, and Z. Wu, "Design of an efficient multi-objective recognition approach for 8-ball billiards vision system," *Kuw3a9it J. Sci.*, no. 45, 2018. Available at: `https://journalskuwait.org/kjs/index.php/KJS/article/view/2083/234`.

[26] A. Gabel, T. Heuer, I. Schiering, and R. Gerndt, "Jetson, where is the ball? using neural networks for ball detection at robocup 2017," in *RoboCup 2018: Robot World Cup XXII* (D. Holz, K. Genter, M. Saad, and O. von Stryk, eds.), (Cham), pp. 181–192, Springer International Publishing, 2019.

[27] E. Schmidt, "Measuring the speed of a floorball shot using trajectory detection and distance estimation with a smartphone camera : Using opencv and computer vision on an iphone to detect the speed of a floorball shot," 2016.

[28] J. Taylor, *Introduction to error analysis, the study of uncertainties in physical measurements*. 1997.

[29] D. M. Allen, "Mean square error of prediction as a criterion for selecting variables," *Technometrics*, vol. 13, no. 3, pp. 469–475, 1971.

[30] A. Alqahtani, M. Zohdy, S. Ganesan, and R. Olawoyin, "A novel phase tracking in zigbee receiver using extended kalman filtering over awgn channel," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, pp. 0158–0162, 2019.

[31] B. Mulgrew and C. Cowan, "An adaptive kalman equalizer: Structure and performance," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, no. 12, pp. 1727–1735, 1987.

[32] "Typical pool ball speeds," Accessed: 05.06.2020.