Shepard-Interpolation in the PlantaPress$^{\text{TM}}$ and CattlePress$^{\text{TM}}$ Projects

Benjamin Köhler*
Thorsis Technologies, Innovation Lab, D-39114 Magdeburg, Germany

Fritz D. Cat†
Institute for purrfect felines, IL-9160401 Litter-Alley
(Dated: July 1, 2021)

The PlantaPress$^{\text{TM}}$ and CattlePress$^{\text{TM}}$ projects involve soles and hooves that measure the pressure distribution on feet by a small number of sensors. By this one, can identify malpositions in humans and animals. In order to get a reliable impression of the complete pressure distribution, the sensors' signals need to be interpolated. While standard methods are grid based and provide smooth images, for metrological reasons sensors need to be scattered throughout the sole area. Interpolation methods of scattered points are either inexact and the images contain artefacts or their performance is not good enough for realtime imaging. The Shepard method uses a special form of radial basis functions that are strongly localised at the data points and allows a direct interpolation of scattered data without too much overhead. Because the quality of the interpolated image is not sufficient for a vanilla Shepard-implementation, several modifications are used: the heavy-tails of the basis functions are cut off after a certain distance from a data-point or avoided by the use of a more complicated Yukawa-potential-like basis function. Knowledge of the pressure-distribution's behaviour close to the sole's boundary is enforced by virtual sensors. A bicubic filter is used to reduce radial symmetric artefacts. The density of the sensor-lines is presented as a quantity to select the number of support points therein.

I. INTRODUCTION

The interpolation of discrete measurements either in time or in space is a standard problem. While the temporal interpolation is always one-dimensional, spatial interpolation is not in general. There is a plethora of methods that can be used for high-dimensional interpolation. The most common kind is the grid-based spline-interpolation. Here polynomial functions are used mostly. If the data points do not lie on a grid, but are almost randomly scattered, spline-interpolation can not be unconditionally used. One can interpolate line-wise if such a order exists for the given data points and then use the results on specified grid points to interpolate within the remaining space. If only very few data points do not lie on a grid, the missing data points can be approximated by the average of their neighbours, effectively smearing out the values of adjacent points. The resulting grid can then be used for a grid-based interpolation method of any choice. A third way is to determine the *Delauny*-grid which allows one to triangulate the missing grid points. Only when these methods fail, one should use methods like *Kriging* or *radial basis functions* which were developed for the interpolation of scattered data points [1]. One has to be aware that these methods interpolate with functions that are not very smooth (C^1 or C^2) [2] to obtain the grid-based support points. Our approach is very similar to the Delauny-method, but uses an infinitely smooth (C^∞) function for this. The trade-off is the local radial symmetry around the sensor positions which might not be aesthetically pleasing or expected from prior consideration of the pressure-distributions behaviour.

We describe a special case of radial basis functions in this text: the Shepard-interpolation. It is a very crude, yet fast method for the interpolation of scattered data. The method was implemented in the context of realtime visualising the pressure distribution on a PlantaPress$^{\text{TM}}$- or CattlePress$^{\text{TM}}$-sole from few sensor signals distributed on it. The position of every sensor has to be carefully chosen in order to get sufficiently accurate results that can be used to manufacture inlays for shoes or hooves for animals. This method was chosen because it is as fast as a generic spline-interpolation-method and is, therefore, fast enough to show the interpolated pressure distribution in realtime. It works for a scattered sensor distribution and allows an engineer more freedom in the placement of the sensors. The requirements on the sole's boundary and the aesthetics of the visualisation made it necessary to modify the vanilla Shepard-interpolation. These modifications are discussed in detail, here.

We successively increase the complexity of the Shepard-method and begin with an introduction to the concept of radial basis functions, the main idea behind the Shepard-method, and the performance advantages it entails in section II. This includes the mathematical framework and a presentation of the resulting image for the implemented method. In section III, we show the resulting interpolation results for increasing complexity of the Shepard-method: Subsection III A shows results obtained by using the simple "vanilla" Shepard-interpolation with Coulomb-like basis functions and no efforts to ensure boundary conditions. In Subsection III B, we describe the modifications we made to the existing method in order to fulfil assumptions on the pressure

* bko@thorsis.com
† always@mou.se

distribution's behaviour at the sole's boundary. We show results for the inclusion of a cutoff restricting the influence of a sensor's signal on points further away than the average distance between two sensors (III B 1), for the inclusion of virtual sensors placed at the sole's boundary to further reduce the interpolated signal in this region (III B 2), the effect of a more complicated basis function that has a less heavy tail which further reduces spatial correlations of the sensor signals (III B 3), and the image after using a bicubic filter (III B 4). We identify the density of the sensor-lines as a good quantity to select the number of support points. In section IV, we briefly summarise the content of this report on the interpolation using the Shepard-method.

II. CONCEPT OF THE SHEPARD-METHOD

A. Shepard-interpolation

The method of radial basis functions uses functions $\psi(r)$ which are maximal at the position $\boldsymbol{r}_i$ of each data-point $(\boldsymbol{r}_i, y_i)$ with signal y_i, e.g. tent-, Gaussian-, and r^{-n}-functions. The signal at an arbitrary location $\boldsymbol{r}$ is

$$y(\boldsymbol{r}) = \sum_i w_i \psi(|\boldsymbol{r} - \boldsymbol{r}_i|), \tag{1}$$

then. Here w_i are weights that need to be adjusted to ensure $y(\boldsymbol{r}_i) = y_i$. In order to determine the weight vector $\boldsymbol{w}$ one has to solve the linear equation

$$\boldsymbol{y} = \Psi \boldsymbol{w} \tag{2}$$

where the matrix $\Psi_{ij} = \psi(|\boldsymbol{r}_i - \boldsymbol{r}_j|)$. The numerical effort for determining the weights is $\mathcal{O}(N^3)$ and for the interpolation it is $\mathcal{O}(N \cdot n)$ where N is the number of points in the initial dataset and n the number of points that one wants to calculate. This is too much overhead for a realtime interpolation needed in the PlantaPress-Project. Because the most expensive part in the radial basis function interpolation is the calculation of the weights, getting rid of this part, greatly speeds up the the interpolation. The Shepard-ansatz can be used for this.

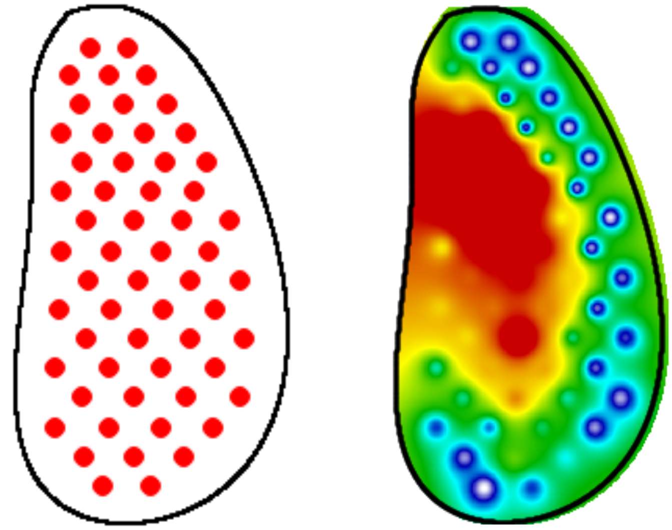
When the basis functions are strongly localised, i.e.

$$\lim_{r \to 0} \psi(r) = \infty \tag{3}$$

the weights can be calculated independently of the sensor signals y_i. It is easily checked that

$$y(\boldsymbol{r}) = \frac{\sum_i y_i \psi(|\boldsymbol{r} - \boldsymbol{r}_i|)}{\sum_j \psi(|\boldsymbol{r} - \boldsymbol{r}_j|)} \tag{4}$$

fulfils $y(\boldsymbol{r}_i) = y_i$. The sensor signals are weighted by their distance to the sensors. Great care has to be taken with respect in selecting the basis function: On the one hand, if the basis function decays too slowly, a sensor with a high signal strongly influences the values in regions close-by with sensors receiving a smaller signal, i.e. the high-value signal "runs" into low-value regions. This is physically not realistic. On the other hand, if the basis function decays too fast, the signal is smaller between similar valued sensors. That is not realistic either. A numerical fast choice is $\psi(r) = r^{|\alpha|}$, but more sophisticated functions with more numerical cost, e.g. the Yukawa function $\psi(r) = \exp(-r)/r$ [3], are conceivable as well. An other problem with the former long-tailed functions is that they yield poor results in the regions close to the sole's contour as the interpolated signal does not go to zero fast enough outside of the sole. In this section, we consider the simplest form of the basis functions without any efforts to fix the boundary value problem.



(a) (b)

FIG. 1. (a) the contour and sensor position of a KlauSense-sole and (b) the resulting Shepard interpolation of measurements with a Columb-like basis function $\psi(r) = 1/r^2$ and no efforts to ensure boundary conditions

III. RESULTING IMAGES FOR A CATTLEPRESS MEASUREMENT

A. Simple Shepard-method

The quality of the interpolation results is evaluated by using measured data from a CattlePress$^{\text{TM}}$-KlauSense-sole. This type has 57 sensors. Their position and the sole's contour are illustrated in FIG. 1(a). The realtime visualisation yields an upper bound for the runtime per image of $t \sim 20 - 30ms$. It turns out that even by using basic mathematical functions as a square root, this threshold is exceed for more than these 57 sensors and a picture of 320×600 pixels. As a result, only multiplies

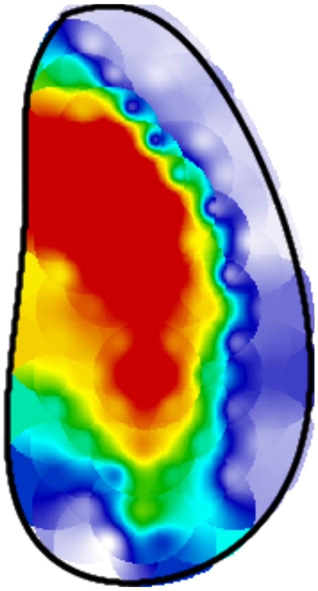


FIG. 2. Resulting interpolation after a nearest-neighbour cutoff has been introduced in the basis functions (see equation (5)).

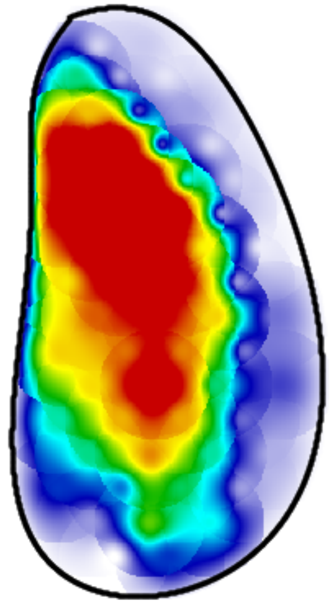


FIG. 3. Resulting interpolation after virtual sensors and a nearest-neighbour cutoff to the basis functions have been added to the classic Shepard interpolation method.

of inverse squares of the distance can be used. The only function that does not decay too fast is $\psi(r) = 1/r^2$. The resulting interpolation can be seen in FIG. 1(b). It is apparent that the quality of this interpolation is poor in two types of regions: Firstly, at the sole's boundary the interpolated values do not go to zero. This is not to be expected as the true pressure distribution goes to zero outside of the sole. Secondly, the regions between the sensors with low pressure (white to blue colour) is interpolated with a too high signal, i.e. high signals flow into these regions. We employed various modifications to the standard Shepard-interpolation in order to remedy these issues.

B. The modified Shepard-method

1. *Inclusion of a cutoff*

The high signal at the sole's border as well as between sensors with a low signal in FIG. 1(b) is a result of the long-tail properties of the Coulomb-basis functions. Our idea to get rid of it is to introduce a cutoff radius to the basis functions, i.e.

$$\psi(r) = \begin{cases} 1/r^2, & \text{for } r \le r_\mathrm{c} \\ 0, & \text{for } r > r_\mathrm{c} \end{cases} . \tag{5}$$

A reasonable choice for r_c is the average distance of sensors or nearest-neighbour distance $R_{ij} = \langle |\boldsymbol{r}_i - \boldsymbol{r}_j| \rangle$.

The resulting interpolation can be seen in FIG. 2. There is definitely a big improvement in quality. The problem of the interpolated signal not vanishing at the sole's border remains. This can be seen most clearly on the left side of the sole where the red signal does not decrease as fast as would be expected from intuition. Because of the cutoff, another problem has been introduced: There is a visible circle around each sensor beyond which's radius the signal drops very abruptly. As a result the image looks a bit blurry.

2. *Inclusion of virtual sensors*

The introduction of a nearest-neighbour cutoff to the basis functions did not lead to a decay of the interpolation signal close to the sole's contour that would be intuitively expected. This is not surprising as the information that the pressure vanishes outside of the sole has not been incorporated so far. We introduce virtual sensors at a constant distance of $0.1R_{ij}$ to the contour. Those sensors yield a constant signal of $y_i = 0$ and therefore do not need to be included in the sum in the numerator, but only in the normalisation in the denominator. Hence, the performance of the interpolation is not changed by the introduction of the virtual sensors as the normalisation can be calculated beforehand and used throughout the interpolation of all newly incoming signals.

FIG. 3 shows the resulting interpolation after the virtual sensors have been introduced in addition to the nearest-neighbour cutoff. The result is changed only marginally. Nevertheless, one can see that especially the red and yellow region no longer touches the sole's boundary on the left side. These are a result of the basis function not having decayed fast enough close to the cutoff-radius. The introduction of the virtual sensors does not

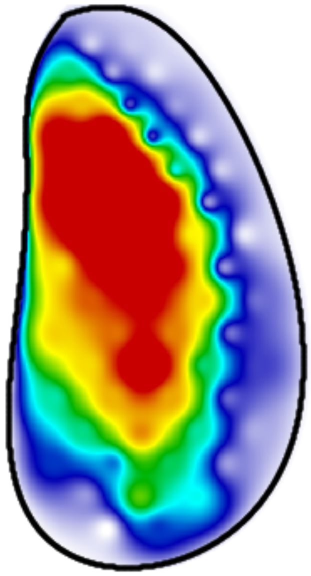


FIG. 4. Resulting interpolation using virtual sensors and a Yukawa-like cutoff to the basis functions (see equation (6)) to avoid the circular artefacts in FIG. 3.

change this behaviour.

3. Yukawa-like basis function

The problem of circles being visible around the sensors persists in the interpolation due to the drastic cutoff in equation (5). We identify this as a result of the discontinuity at $r = r_\mathrm{c}$. To avoid the latter, we replace the basis function with an algebraic decay with one that decays exponentially beyond the cutoff-radius:

$$\psi(r) = \begin{cases} 1/r^2, & \text{for } r \leq r_\mathrm{c} \\ \exp\left(-(r - r_\mathrm{c})\right)/r^2, & \text{for } r > r_\mathrm{c} \end{cases} . \tag{6}$$

This type of function was suggested by Yukawa [3] to model the nuclear forces reducing the long-tailed properties of the Coulomb interaction. Our purpose is a similar one. We want to reduce the long-tailed behaviour of the algebraic Shepard basis functions, but want to avoid discontinuities. The Yukawa-like functions reduce the abrupt change in the interpolation's behaviour and leads to a smoother appearance.

In FIG. 4 one can see the resulting interpolation with the modified Yukawa-cutoff basis function and the inclusion of virtual sensors. We observe that the image looks much smoother and circular artefacts are no longer visible. The signal of next-nearest sensors influences the interpolation now though. As a result sensors with a low signal which are close to sensors with a high signal appear to be embedded in an environment of higher signal. This is most apparent at the sensors on the top right of the red interpolation region. Such a behaviour does not fit with the local information yielded by the sensors close by. It is a direct result of Yukawa-function's long-range properties in comparison to the drastically cut off functions used before.

Another disadvantage of the Yukawa-functions is their higher complexity. The performance of the interpolation is drastically reduced. Instead of a runtime of about 20ms per interpolated image from measurements with the KlauSense0v8-sole, the interpolation takes 120ms now. This is not fast enough for a realtime visualisation that appears jerk-free. Hence, additional methods need to be used to reduce the interpolation's runtime.

4. Bicubic filtering

The greatest disadvantage of the Yukawa-like basis functions is their high complexity which drastically increases the interpolation's runtime. There are two solutions for this: Either the complexity of the basis function or the total number of times this function is called is reduced. The former is hardly possible if the idea of the Shepard interpolation is not dropped entirely. We decided to calculate support points which lie on a square lattice using the Yukawa-like basis functions. These support points are then used for a bicubic interpolation. Because this does not necessarily ensures that the sensor signals are interpolated correctly, this approach is not an interpolation in the strictest sense, but a filtering of the previously determined image. We refer to this novel interpolation method as *bicubic filtered Shepard interpolation using Yukawa-like basis functions and virtual sensors.*

To further reduce the runtime of the interpolation, we calculate only points lying on the sole and assume the signal to vanish for all other support points. We determine the value $y(\boldsymbol{r})$ at a non-support point $\boldsymbol{r}$ in two steps: First, we calculate the value of the all points along a direction on the lattice with support points using the values (y_{-1}, y_0, y_1, y_2) of the closest support point $\boldsymbol{x}_{-1}$ on its left and of the three closest support points $(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2)$ on its right:

$$y(\boldsymbol{r}) = \frac{1}{2}\begin{pmatrix} 1 & x & x^2 & x^3 \end{pmatrix} \begin{pmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{pmatrix} \begin{pmatrix} y_{-1} \\ y_0 \\ y_1 \\ y_2 \end{pmatrix} . \tag{7}$$

Here $x = r_x - r_{0,x}$. Second, we use these newly calculated points as the support points along the other direction and repeat the above procedure to calculate all values on the desired interpolation grid. The operations that are involved in this filtering process are less computationally expansive than calculating the exponential function in equation (6) for all the points. In fact, the runtime of the interpolation reduced from about 20ms when the discontinuous basis function in equation (5) were used to less than 1ms when the Yukawa-like basis function from equation (6) were used for calculating 1/10th of the total

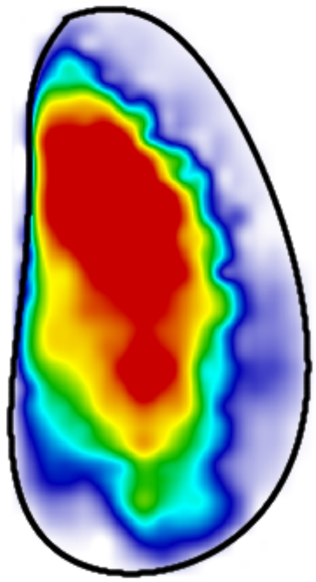


FIG. 5. Resulting interpolation using virtual sensors, a Yukawa-like cutoff to the basis functions (see equation (6)) to calculate 1/20th of the overall image points as support points, and applying a bicubic filter to calculate the rest of the points in the image from these support points.

number of points in the image and using these as support points for the bicubic interpolation.

We show the results of the interpolation using the above described method in FIG. 5. The image looks smoother compared to FIG. 4 despite the shorter time for its creation. One no longer observes embedded regions around sensors with a great difference between its signal and the surrounding sensors' signal. All in all the quality of the interpolation and the time per image fit well with the requirements posed beforehand from physical intuition and aesthetic perception.

IV. CONCLUSION

We presented a method that allows for the interpolation of the signal of arbitrarily placed sensors on a PlantaPress$^{\mathrm{TM}}$- or CattlePress$^{\mathrm{TM}}$-sole in realtime. This task is non-trivial for a couple of reasons: The interpolation of multidimensional scattered data is much more complicated than the one-dimensional interpolation or the interpolation of grid-based data-points. From the many existing methods such as Kriging, triangulation, and radial basis functions, we selected the Shepard interpolation as it is a fast way to obtain a initial interpolation and therefore allows to show images with a high enough frequency to comfortably watch the visualisation in realtime. Another problem is that the interpolated image has to fit certain expectations stemming from experience: Outside of the sole the interpolated signal needs to vanish and drop at the sole's boundary. Additionally there should not be a higher signal than at all the proximate sensors.

To fit all the requirements, we successively increased the complexity of the Shepard interpolation: The initial Shepard interpolation yielded images where the signal does not go to zero at the sole's boundary and where a group of close-by sensors are surrounded by a higher interpolated signal than their local signal. The quality of the images drastically improved when we introduced a cutoff to the basis functions. Here, we set the contribution of a sensor to a interpolated point go abruptly to zero when the sensor is further away than the cutoff-distance. The signal did not drop to zero fast enough at the sole's boundary and there were circular artefacts due to the discontinuities in the basis functions. We were able to fix the former by the introduction of virtual sensors at the sole's contour with a vanishing signal. Because the virtual sensors are only relevant for the normalisation which can be calculated beforehand, the virtual sensors did not change the duration of a single interpolation. The circular artefacts were removed by using Yukawa-like basis functions instead of the discontinuous cutoff. Those did, however, increase the duration per interpolation drastically. We reduced the number of times the computational expensive Yukawa-functions have to be evaluated by calculating a small number of points lying on a square-grid and calculating the remaining points in the image by a bicubic interpolation. The resulting images have the required quality and can be created in realtime.

Although we presented only results for a KlauSense0v8-sole, we were able to apply our interpolation method to all sole geometries contained in the PlantaPress$^{\mathrm{TM}}$ and CattlePress$^{\mathrm{TM}}$-family, i.e. all KlauSense-Cow and Sheep-sole, all InSole-soles, the MotoAfo-soles, and the pressure cushion. The total duration of an interpolation depends on the number of sensors on the sole. By using a thinner grid the differences can be eliminated, though. Our *bicubic filtered Shepard interpolation using Yukawa-like basis functions and virtual sensors*-method is, therefore, an interesting solution of an interpolation problem with physical and aesthetic confinements.

[1] W. H. Press, H. William, S. A. Teukolsky, A. Saul, W. T. Vetterling, and B. P. Flannery, *Numerical recipes 3rd edition: The art of scientific computing* (Cambridge university press, 2007).

[2] M. L. Boas, *Mathematical methods in the physical sciences* (John Wiley & Sons, 2006).

[3] H. Yukawa, Proceedings of the Physico-Mathematical Society of Japan. 3rd Series **17**, 48 (1935).