PART IV
Interacting Floor

December 2000, Simon Bovet

1 Introduction

The purpose of this essay is to investigate the possibilities of some interaction between humans and a floor made out of weight-sensitive and light coloring tiles. This is closely related to the ADA Intelligent Space Expo.02 project since it is dealing with the concept developed for this purpose of sensitive and coloring floor.

The input to the floor consists of the information from the weight-sensor of each tile (i.e. whether someone is standing on it or not), and the color in which each tile is lighten shall be seen as the output from the floor. Interaction between humans and floor is meant as soon as the response of the floor adapts to the various behaviors of the persons standing on the floor, which are also somehow reacting to the visual output from the tiles.

For this purpose, I developed a network simulator in order to better see and appreciate the possibilities of such a floor.

In the first part, I discuss the physical restriction of the network as well as some techniques enabling the floor to retrieve maximum information from the persons moving around on the floor. In a second part, I present a few ideas of reactions the floor can show up, as well as ways the floor can respond and adapt to the human behavior of the persons. In the last part, I discuss briefly a few results and measurements done with the simulator.

2 Description of the simulator

2.1 The data

2.1.1 Network architecture and restriction

The network consists of an array of cells (hexagonal tiles), each being able to provide some input, generate some output and communicate (i.e. share or peek information) only with the six neighboring cells. We impose therefore a local and isotropic behavior, which has the great advantage of not depending on the size or the shape of the array (which can therefore be easily modified or extended). The only exception is done for a very few global data that are shared with the floor itself, such as the time value (generation number) or the output patterns.

The steps are discrete in time. At each generation, every tile computes sequentially its state (i.e. the value of the different parameters) for the next generation, pretty much like a cellular automaton. For a few parameters, a longer history is kept for the values of the previous generations.

2.1.2 Input

The only input to the floor collected by each cell is a boolean value indicating whether someone is standing on the corresponding tile or not (e.g. by the way of some weight sensor threshold).

2.1.3 Output

The only output from the floor is the color that each tile can be illuminated by. Since each tile is equipped with three different color-lamps which intensities can be varied, a large color spectrum is available for the output. In the simulator, it can be used to indicate various cell parameters (distance, density, etc.). But the real output that is meant to be used with humans are mainly the patterns (such as an arrow to indicate the direction of an invitation) according to which a group of tiles is colored.

2.2 The humans

The simulator does not only simulate the floor, but also the humans moving around on it. Each person is described by a set of parameters, such as:

- Position, speed and direction[1] (as function of time)
- Acceleration[2] probability, direction change (*deflection*) probability
- Maximum speed, acceleration, deceleration and deflection.
- Invitation attention probability and response.

At each generation, each human position is updated as well as the sensor value of the cell(s) underneath it. There is no restriction on the number of cells for which the sensor value is activated for one person, as it may be standing across two or more cells.

The response of a human to a floor output is described with the last parameters above. At each generation and with a probability given by the *invitation attention probability*, each human looks at the color of the closest tile in a given random direction. If it is not the default color, then it changes its speed direction towards it. This deflection amount is weighted with the *invitation response* parameter.

The position of each person is confined in an area enclosing the floor. If one person is moving out of it, either it is elastically reflected at the boundaries or the person is removed, with a certain probability (*floor permeability*).

2.3 Getting information from the humans

The first goal of the floor is to get contact with and perceive its environment, namely the humans. That is why it has to retrieve as much information as possible from its input, such as trying to locate, track and discern the persons and describing the way they are moving on the floor by discerning their motion, their relative position to each other, etc.

---

[1] In this context, the *speed* is the norm of the velocity vector and *direction* is its direction.

[2] Change of speed between two time steps (discrete derivative of speed)

2.3.1 Person tracking

This is probably the most critical task, as one person can get on the floor, leave it, stand on several tiles simultaneously, jump from one tile to another, or even not activate any tile sensor for one or more generation (if it is "running" or standing on several tiles with its weight not large enough to activate the sensors). Person *crossing* must also be considered and solved in some way (see 2.3.2).

This is done with the following method: each cell contains two variables (in addition to the *person identification number*), the *number of person on*, and the *number of person lost*. These are real numbers, as we want the floor to know when a person is standing on several tiles simultaneously (e.g. if someone is standing on two tiles, the number of person on should be one half for both). The following algorithm defines these two variables:

- If the sensor value is set to on (i.e. was off the previous generation):
  - ☛ Set the new number of person on to the last number of person lost (persons lost are back) and the new number of person lost to zero.
  - ☛ If the number of persons on and lost for all neighboring cells is zero, create a new person for the cell (i.e. get a new unique identification number and set the new number of person on to one).
  - ☛ If someone is on or lost on some neighboring cell(s), share information with it, i.e. get same identification number (the same person is probably moving from the neighboring cell onto the current one), set number of person on to the number of person lost (persons lost are back). Finally average the number of person on with the neighboring cell.
- If the sensor value is set to off (i.e. was on the previous generation):
  - ☛ If nobody is around (number of persons on and lost for neighboring cells is zero), set the number of person on to the number of persons off (persons on are lost).
  - ☛ If someone is around cast equally the number of person on of the current cell to the neighboring ones.
- If the sensor value do not change:
  - ☛ If someone is on the cell (number of person on greater than zero) and that neighboring cells have person lost with the same identification number, bring the person lost of the neighboring cell to the person on of the current cell.
  - ☛ Otherwise, simply average the number of person on and lost with the value of the neighboring cells.

So far, the persons are tracked when they are moving on the floor. Figure 1 and Figure 2 show two examples of this algorithm.

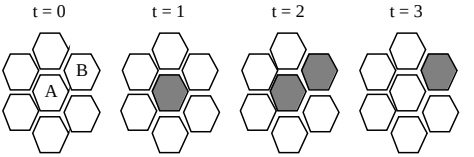


Figure 1: Six cells at four successive generations. A cell in gray indicates that the weight sensor is on.

	t = 0	t = 1	t = 2	t = 3
Identification number (A)	-	1	1	-
Person on (A)	0	1	0.5	0
Person lost (A)	0	0	0	0
Identification number (B)	-	-	1	1
Person on (B)	0	0	0.5	1
Person lost (B)	0	0	0	0

Table 1: Cell parameters according to Figure 1.

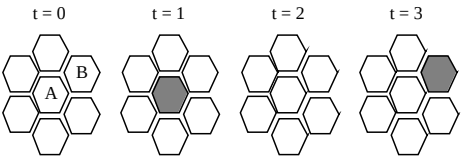


Figure 2: Six cells at four successive generations. The person was not detected at the third generation.

	t = 0	t = 1	t = 2	t = 3
Identification number (A)	-	1	1	-
Person on (A)	0	1	0	0
Person lost (A)	0	0	1	0
Identification number (B)	-	-	-	1
Person on (B)	0	0	0	1
Person lost (B)	0	0	0	0

Table 2: Cell parameters according to Figure 2.

2.3.2 Detecting person crossing

The problem is now to detect when two (or more) persons are crossing. If two persons are standing on the same tile, there is no way of distinguishing them before they move apart. Let us investigate the configurations described by Figure 3.

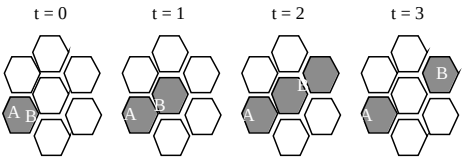


Figure 3: Two persons are moving apart after standing on the same tile.

At the generation t = 0, only one tile is being stood on, and the floor interpret that as only one person being present. At the next generation, the previous algorithm interpret that this person is now standing on two tiles, as well as for the third generation t = 2 where the person is extending on three tiles. Only at the last generation t = 3, there is a clue that two person where actually on the floor and that they are moving away from each other.

The following algorithm gives a solution for that problem:

- If a tile is "left" by someone, i.e. the sensor value switches from on to off (as in the case of the middle tile for the last two generations of Figure 3), and if the neighboring tiles with some person on are not next to each other (as the upper right and lower left tiles), then *mutate* the neighboring cells (all but one, so that it also works with three persons moving apart simultaneously). *Mutating* a cell means setting the *mutation parameter* of the cell to a new unique person identification number.
- When idling, each cell propagates the mutation to its neighbors in the following way:
  - ☛ If no neighboring cells with the same person identification number are mutating, then propagate the mutation (i.e. set the mutation parameter of all neighboring cells with the same person identification number with the same value) and mutate (i.e. change its own person identification number to the value of the mutation parameter).
  - ☛ If a neighboring cell is mutating with another mutation parameter value, then just adopt the new mutation value of the other cell.

One could think of simply changing the identification number of one neighboring cells instead of involving a whole mutation process, but problems would then arise if the persons are extending on several cells or in some other marginal cases.

2.3.3 Distance to nearest persons

One useful piece of information is to know, for each cell, how far the *n* nearest persons are. This can be achieved with the following algorithm:

Each cell contains a list of *n* elements, each element being either empty or containing parameters such as (see 2.3.4 for further parameters):

- The person identification number.
- The distance of the corresponding person.

This *nearest list* is always sorted by the distance value (for some internal purpose) and is updated as follow:

- Build a new list according to the nearest list of each neighboring cell.
  - ☛ If someone is standing on the tile, insert a new element with the identification number of the person and with distance set to zero.
  - ☛ For each element of all neighboring nearest lists,
    a) insert the element into the new list if no element already exists with the same identification number or
    b) replace the distance value of an existing element with the same identification number but with a higher distance value.
  - ☛ Increment the distance of each non-empty element by one unit (we are *propagating* the distance one cell further, so that the distance has to be increased).
  - ☛ Sort the new list, which now contains all nearest persons with the smallest distances.
- Copy the *n* first elements of the above-created list to the new nearest list of the current cell.

2.3.4 Person velocities

Another useful information that the floor can retrieve is the speed at which the humans are moving. To be exact, we want to know at what velocity the *n* nearest persons are moving towards or away from a cell. This could simply be done by taking the difference of distance between two generations. Unfortunately, some problems arise because of the way the distance information propagates (the distance value is only accurate after the *distance field* has relaxed, i.e. after a few generations for some direction).

That is why a *phase* information is included in the parameters for the *n* nearest persons. The *phase field* propagates the same way the distance field does, except that the initial phase value generated by a person is steadily increasing by one unit (whereas the distance value for a cell being stood on is always set to zero). The result is that the phase field describes some wave propagating with the *speed of light*[3] around each person.

Reading the value of one of the *n* phase fields of a cell (one cell contains information about the *n* nearest persons to it) provide sufficient information to calculate the speed at which the person is moving towards the cell. The phase φ is increasing with generation number *t* and is delayed by *d*, where *d* is the distance between the person and the cell. We have therefore

[3] The speed of light in this context is the maximum speed at which information can be propagated in the floor, namely one cell distance per generation time.

$$\varphi = \varphi_0 + t - d$$

The (discrete) derivative with respect to t gives us the following relation for the speed v (*Doppler effect* with speed of light = 1):

$$v = \frac{\Delta d}{\Delta t} = 1 - \frac{\Delta \varphi}{\Delta t}$$

(For some better results, each cell keeps a history of the m last nearest person parameters, so that $1 \le \Delta t \le m$).

2.3.5 Local person density

The *local person density* ρ_L of a cell is also a useful parameter, which will be used by the interaction (see 3.1.2). It is defined by

$$\rho_L = \sum_{k=1}^{n} \frac{1}{d_k^2},$$

where the d_k is the distance to the k-th nearest persons (see 2.3.3). It corresponds to the discrete approximation of

$$\rho_L(x) = \int \frac{\rho(x-y)}{|x-y|^2} d^2y, \quad \rho(x) = \sum_{k=1}^{n} \delta^{(2)}(x - x_k)$$

which has the dimensionality of a density (number of person per area).

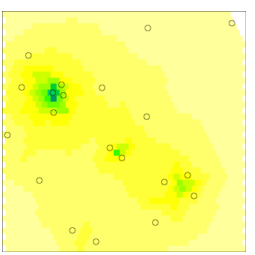


Figure 4: Local density for a given configuration. The persons are indicated by circles, and the color indicates the local density value (light color for low density value).

3 Floor reactions

So far, the floor is able to retrieve some piece of information about the humans on it, but there is still no interaction. The proposed interaction scheme is divided into two parts: the

first is to generate some output indicating to the persons that the floor want to interact with them, the second is to interpret their reactions (or non-reactions) in order to adapt to their behavior.

3.1 Generating output

3.1.1 Gravitation

This is a first attempt to generate interaction between humans. It is rather complicated to implement compared to the second method (see 3.1.2) and give no significantly better results. But from a quantum-physical point of view on this floor "universe", it comes out as being a rather natural implementation of Newton's law, and is therefore worth a few explanations.

The attraction the floor wants to create between two persons relies on some exchange of particles. These *gravitons* are particles moving with speed of light in one parameter space of the cells. They are created in the cells for which the two nearest persons are equally distant. As they are propagating, they accumulated some *action* according to their path or other parameters of the cells they are travelling through. A cell on which someone is standing, reacts to the gravitons by applying a *force* (see 3.1.3) in the direction from which the graviton of *least action* is coming.

The action of a graviton is increased when its propagation direction is changing (*kinetic term*). At each cell through which the graviton is propagating, the distance value of the nearest person is added to the action (*potential term*). These are just a few examples, but one could think about including a coupling with the speed information of the cell, in order to avoid two persons being brought together in the way of other persons moving about.

3.1.2 Local density limits

The *local density interaction* may be more familiar and natural. It is in any way much easier to implement than the previous one. It is based on the idea that people vary in their wish to be close to each other. More social persons will tend to stay in group, whereas solitary persons may prefer to stay alone.

The initial purpose of this interaction was to control the human flow on the floor, e.g. in avoiding people overcrowding some area or in recalling persons moving in uninteresting directions. This is why the concept of *local density* has been implemented above (see 2.3.5).

For each person[4] there are two threshold values, corresponding to the *minimum density* and *maximum density*. If the person is standing on a region where the local density is less than its minimum density value, the floor will apply a *force* (see 3.1.3) on the person in the direction of the *density gradient* in order to invite him to return to some higher density region. On the other hand, if the local density where the person is standing is greater than its maximum

[4] These parameters (as well as all other parameters that are *attached* to a person) are actually contained in each cell being stood on, and are propagated to other cells being stood on when the person is moving.

density value, the force will be applied in the opposite direction of the density gradient, so that the person is repelled to some lower density area. See Figure 5 for an example.

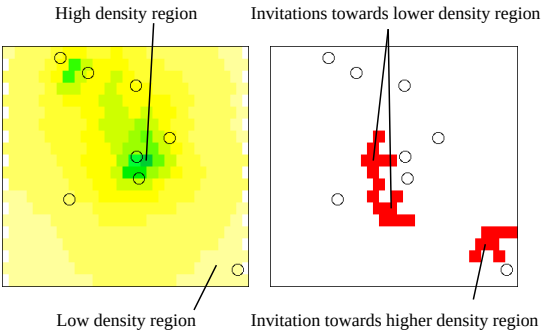


Figure 5: Local density (left) and tile colors (right) for a given configuration of persons.

3.1.3 Force and invitation

When the floor wants to invite someone to move in a given direction, the only means available is to light the tiles around the person with some pattern, e.g. by "drawing" an arrow (see Figure 5) or by making a tile blink.

If the tiles are immediately lighten as soon as the floor wants to invite someone, the result is unstable as the person can move or the parameters vary. The patterns may not have finished drawing, as the floor already wants to invite the person in another direction.

In order to add some reaction delay or inertia, the invitation is done by the way of a *force* concept. When the floor wants someone to move in a direction, it applies a force on the person (i.e. increase the force value for the given direction). As soon as the force value of one person for one direction is above a threshold value, it starts drawing the invitation pattern. Because these force values are relaxing (i.e. decreasing at each generation), the floor has to apply a force for a few generations in the same direction before the invitation is shown, which is exactly the required inertia.

3.2 Adaptation to human behavior

The last step to be done in order to close the "loop" of interaction between the humans and the floor, is the adaptation capability of the floor to the human behavior.

We want e.g. the floor to adjust the various parameters to best meet the behavior of the persons, or to interpret the human reaction when a person is responding to a visual invitation.

This is done mainly for the local density limit parameters, but it can easily be extended to some other reactions from the floor.

3.2.1 Idling

The goal is to adjust the two density threshold parameters of a person, namely the maximum and minimum density thresholds (see 3.1.2). The way it is done is the following:

- When a person is standing on a cell where the local density is between the minimum and maximum threshold values, "squeeze" the thresholds by increasing the minimum and decreasing the maximum.
- When a person is standing on a cell where the local density is above the maximum threshold value (under the minimum value), increase the value of the maximum threshold (decrease the minimum).

In both ways, the thresholds will *adapt* to the density where the person is standing. See Figure 6 for an illustrated example.

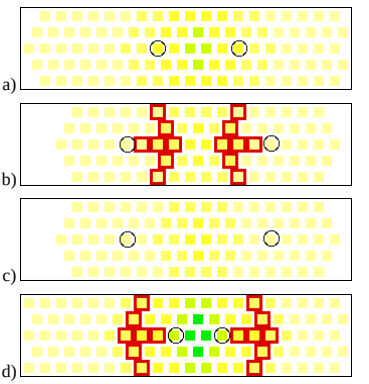


Figure 6: Density threshold value adaptation. The center color of each cell indicates the local density, and the boundaries color the actual output color. a) Two persons are standing on the floor, where the local density is between the threshold values, so that there is no *invitation*. b) The two persons are moving apart. As soon as the local density is below the minimum threshold value, the floor tries to bring them back together. c) After a while, the minimum threshold value of each person has *adapted* and the invitations cease. d) Since the persons have been standing for the last generations on low-density regions, both threshold values have decreased. As soon as they are sufficiently close together so that the local density is above the maximum threshold value, the floor invitations try to bring the apart.

3.2.2 Responding to human reaction

Another information may be used to adapt these threshold values, namely when a human respond to the visual invitation of the floor. The floor detects such a response when a tile of the visual invitation pattern is being stood on (see Figure 7).

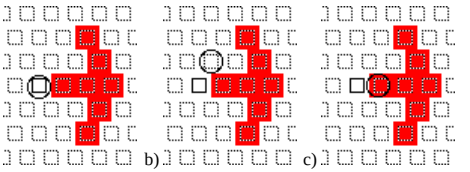


Figure 7: From a) the starting configuration: b) the person is not responding to the invitation, c) the person is responding to the invitation.

In this case, the corresponding density threshold has to be adjusted: if the floor was inviting the person to move to a higher density region (the local density was below the minimum threshold), increase the minimum density threshold (the same is done the other way round).

4 Results

4.1 Entropy of a configuration

In order to describe the behavior of the humans as function of the different parameters, we need a quantity that can measure in some way the randomness of a configuration. The *entropy* of the human distribution over the floor may be a good choice since it measures how probable a certain configuration is, if the humans were to be placed randomly.

The entropy is calculated in the following way. The floor area is first subdivided into a grid with k cells. Then, all numbers n_i of persons contained in the i-th grid cell are calculated as well as the total number of persons $n = \sum_{i=1}^{k} n_i$. The entropy is then given by:

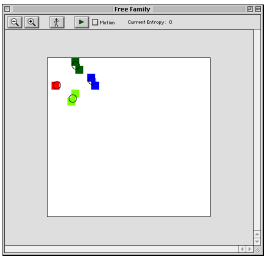
$$S = \frac{1}{N} \sum_{i=1}^{k} \frac{n_i}{n} \ln\left(\frac{n_i}{n}\right)$$

where the normalization factor N is chosen so that $0 \le S \le 1$, and $S = 1$ for $n_1 = \ldots = n_k = n/k$:

$$N = \sum_{i=1}^{k} \frac{1}{k} \ln\left(\frac{1}{k}\right) = -\ln k$$

4.2 Starting configurations

Figure 8 shows the two starting configurations used in this chapter. The first one is composed of four persons, two of them being slower, the two others more attentive to the visual invitations from the floor (it can be seen as a family with two parents and two children). The second configuration is composed of two couples situated in opposite corners.

a) 

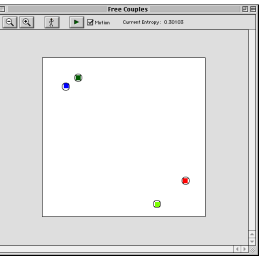
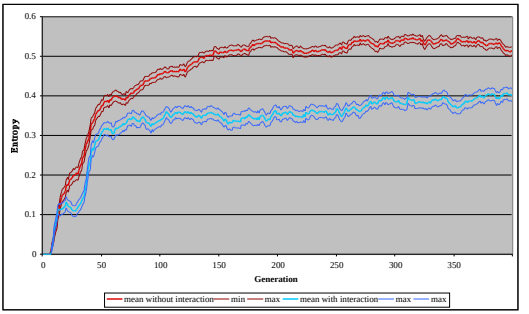
b) 

Figure 8: a) First and b) second starting configurations. The circles indicate the persons, and the tile colors indicate the person identification number.

4.3 Time evolution of the entropy

For both configurations, the simulation is run over 400 generations, and the area is divided into a 3 by 3 grid for the entropy measurement at each generation. 100 simulations are done in both cases, from which the mean entropy and standard deviation is calculated for each generation. The results are shown in Figure 9.

a)



b)

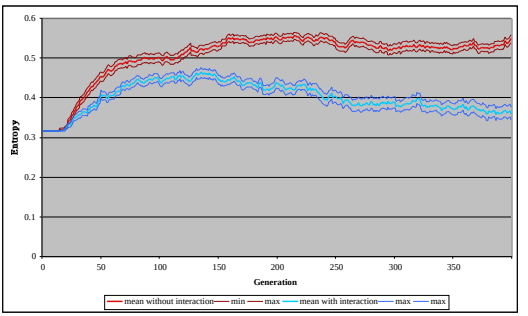


Figure 9: Entropy evolution for a) the first and b) the second starting configuration. The two sets of curves indicate the entropy evolution with and without interaction. The thick lines indicate the averages and the thin lines indicate the standard deviations calculated from 100 different runs.

4.4 Mean entropy

The mean values of the entropy measured from the 50th generation are the following:

	Without interaction	With interaction
First configuration	0.50 ± 0.04	0.36 ± 0.02
Second configuration	0.53 ± 0.02	0.41 ± 0.03

Table 3: Mean values of entropy.

5 Conclusion

We can see from the results on the entropy measurement that the interaction causes an entropy decrease. This means that the visual invitations from the floor affect the human behavior, as the persons are moving "less randomly" than they would without interaction.

This result is though to handle with care. First of all, because the measure of entropy is of little signification as any trivial invitation procedure would produce such a decrease (e.g. by inviting all the persons to move in one fixed direction).

Secondly, because the model use to simulate the human motion is too simple to describe really interesting behavior (such as the increase of interest to play with the floor, or the simple interaction between only humans, etc.).

Then, because it is difficult to measure other significant parameters that are not directly included into the simulation model. The current model is only dealing with the motion of the persons, so we should not be much surprised to see changes in the "motion behavior" as this is precisely what the whole thing is aiming to.

Nevertheless, it is remarkable to see that many piece of information can be gained from a so little amount of input (namely, just the weight threshold) and with the imposed architecture restrictions (locality and isotropy of cell interaction). The possible number of different interaction forms is growing exponentially with the number of parameters that are put into the model!

I think that the simplicity of this model makes it quite interesting and valuable for the quest of better understanding complex systems trying to describe some human behavior.

6 Annex

This annex contains the listing of the code for the main classes of the simulator. All the interface implementation of the program is not presented here because of its negligible interest.

The most interesting class is the CCell class, which implements the main features of a single cell. Some features related to a cell are contained in separated classes for clarity (CNearest, CInvitation, CGraviton). The simulation of the humans is described by the classes CHumans, CPerson. The trivial human-floor interaction (i.e. setting the mSensorValue parameter for each CCell on which a person is standing) is not presented here.