Описание фильтра частиц (Particle Filter)

Фильтр частиц – это фильтр, основанный на принципе Монте Карло, в нашем случае, для решения пробем оценки положения пользователя с применением информации от трех источников (маячки, сенсоры устройства (IMU алгоритм) и информация карты). Суть фильтрации сводится к определению апостериорного положения частиц и расчета их метрики на каждой итерации. Под метрикой следует понимать некий параметр, харакетризующий апостериорное распределение координат с учетом их весов.

Процесс фильтрации состоит из Инициализации фильтра и трех основных шагов (Приложение А). Рассмотрим каждый из этапов по отдельности.

Старт алгоритма - «Засеивание» (первая итерация)

Существует две стратегии для генерации первичного расположения частиц (процедура засеивания).

Первая стратегия применяется в случае, когда информация о первичном расположении устройства неизвестна. В этом случае частички равномерно рассеиваются по всей карте местности и в начальный момент времени имеют равные веса.

Вторая стратегия заключается в том, что существует некая априорная информация о положении устройства, в нашем случае координаты, полученные от маячков в первый (возможно и не в первый) момент времени. В этом случае можно более точно определить область засеивания. В нашем случае ожидаемая ошибка измерения по маячкам составляет ±α м. Поэтому в начальный момент времени частички рассеиваются согласно формуле:

$$[x_i;\ y_i] = [x_{beac} \pm \Delta x_i;\ y_{beac} \pm \Delta y_i] = [x_{beac} + rand(\alpha);\ y_{beac} + rand(\alpha)]$$

где $rand(\alpha)$ – генератор случайного вещественного числа с равномерным распределением в диапазоне от [-α; α].

Блок-схема работы данного этапа алгоритма приведена в Приложении Б.

Шаг 1: Расчет новых координат частиц на основе предыдущих координат с применением модели процесса и данных алгоритма IMU.

Аналогично фильтру Калмана, фильтр частиц требует модели процесса, который подлежит фильтрации. В нашем случае, под моделью процесса будем понимать модель вида:

$$\begin{cases} \tilde{x}_i = \tilde{x}_{i-1} + \Delta x_i \\ \tilde{y}_i = \tilde{y}_{i-1} + \Delta y_i \end{cases}$$

где $[\tilde{x}_i;\ \tilde{y}_i]$ – координаты устройства для i-го момента времени, рассчитанные с помощью фильтра частиц; $[\Delta x_i;\ \Delta y_i]$ – величины приращений координат в направлении осей X и Y для i-го момента времени (определяется при помощи алгоритма IMU, т.е. при помощи сенсоров); $i \in [0;\ I]$ – индекс момента времени.

Данная система уравнений записывается для каждой частицы фильтра (т.е., например, если есть 10 частиц, то будет 10 систем уравнений) с учетом возможного отклонения приращений координат от истинных значений:

$$\begin{cases} x_n^i = x_n^{i-1} + \Delta x_{i-1} + \vartheta_n^X \\ y_n^i = y_n^{i-1} + \Delta y_{i-1} + \vartheta_n^Y \end{cases}$$

где $n \in [1;\ N]$ – порядковый номер частицы; $[x_n^i;\ y_n^i]$ – координаты положения n-ой частицы для i-го момента времени; ϑ_n^X и ϑ_n^Y – случайные величины, которые характеризуют неточность в определении соответствующей координаты каждой частицы (описываются гауссовым законом с нулевым мат. ожиданием и СКО σ_X и σ_Y соответственно; величины σ_X и σ_Y определяются точностью модели, описывающей процесс (по умолчанию равны 0,1 м));

Если модель достаточно точная, то величины ϑ_n^X и ϑ_n^Y незначительно влияют новые значения координат частицы. Напротив, если модель очень неточная либо ее параметры не известны, то параметры ϑ_n^X и ϑ_n^Y (СКО) следует выбирать большими.

Блок-схема работы алгоритма на шаге 1 приведена в Приложении В.

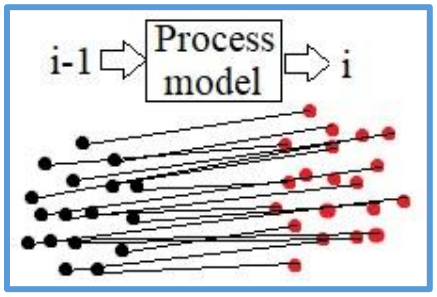


Рисунок 1 – Схема перехода частиц для соседних временных отсчетов

Шаг 2: Расчёт априорных значений весов частиц на основе информации о карте помещения и информации от маячков.

При расчете априорных значений весов частиц учитывается информация карты. Карта условно может быть разделена на белые (допустимые для нахождения или прохождения) и черные (запрещенные для нахождения и прохождения) области (маска карты). Если при переходе из положения (i-1) в положение i частичка пересекает либо оказывается в черной области, то данное перемещение считается невозможным и вес данной частички приравнивается к нулю (вес рассчитывается для каждого уравнения модели отдельно). Т.е.

$$if\ \omega_{nX}^{i} = 0 \mid \omega_{nY}^{i} = 0\ then\ \omega_{n}^{i} = \omega_{nX}^{i} \cdot \omega_{nY}^{i} = 0$$

где ω_{nX}^{i} и ω_{nY}^{i} – веса n-й частицы по координатам X и Y соответственно для i-го шага времени; ω_{n}^{i} – общий вес n-й частицы для i-го шага времени; $n \in [1; N]$ – порядковый номер частицы; N – общее количество частиц.

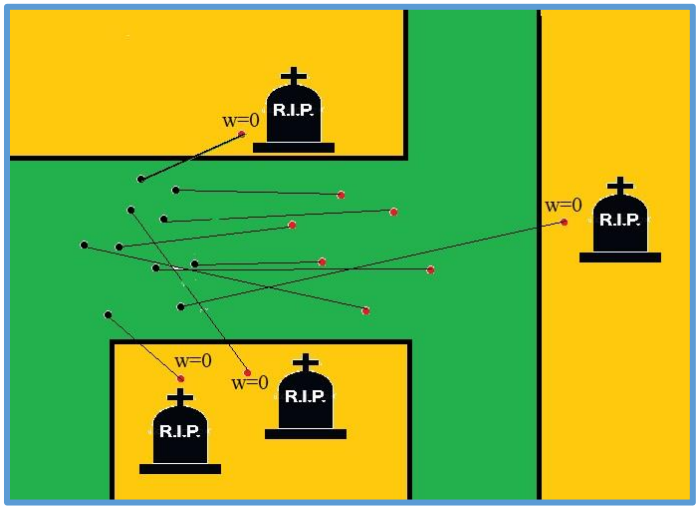


Рисунок 2 – Пример расчета весов частиц

Вес «выживших» частиц (все частицы, которые не пересекали запрещенные области и не находятся в них) рассчитывается по следующей формуле:

$$\begin{cases} \omega_{nX}^{i} = \dfrac{1}{\sigma_X\sqrt{2\pi}} e^{-\frac{(x_n^i - x_{beac})^2}{2\sigma_X^2}} \\ \omega_{nY}^{i} = \dfrac{1}{\sigma_Y\sqrt{2\pi}} e^{-\frac{(y_n^i - y_{beac})^2}{2\sigma_Y^2}} \end{cases}$$

где $[x_{beac}\,;\, y_{beac}]$ – координаты устройства, определенные с помощью алгоритма позиционирования по маячкам.

$$\omega_n^i = \omega_{nX}^i \cdot \omega_{nY}^i = \frac{1}{\sigma_X\sqrt{2\pi}} e^{-\frac{(x_n^i - x_{beac})^2}{2\sigma_X^2}} \cdot \frac{1}{\sigma_Y\sqrt{2\pi}} e^{-\frac{(y_n^i - y_{beac})^2}{2\sigma_Y^2}}$$

ВАЖНО: Как видно из вышеизложенного, показания маячков используются ТОЛЬКО для расчёта весов, поэтому, вполне достаточным будет получение точности положения пользователя по маячкам около 2 метров.

После расчета веса каждой частицы, все значения ω_n^i должны быть пронормированы следующим образом (включая «невыжившие» частицы):

$$\bar{\omega}_n^i = \frac{\omega_n^i}{\sum_{n=1}^{N} \omega_n^i}.$$

Блок-схема работы алгоритма на шаге 2 приведена в Приложении Г.

Шаг 3: Генерация нового поколения частиц из частиц предыдущего поколения с наибольшим весом при помощи алгоритма «Колесо отбора» ("Resampling wheel").

Частицы с нормированными весами подвергаются специальной обработке (отбору) для генерации нового поколения частиц. Для этого существует большое количество подходов, среди которых наиболее простым и эффективным является отбор по методу «Колеса отбора».

Алгоритм «Колесо отбора»

В алгоритме отбора принимают участие все частицы с нормированным весом больше нуля (допускается использовать все частицы с предыдущей итерации, но отбор в этом случае займет больше времени). Вначале определяется максимальный вес частицы, $\max_n \bar{\omega}_n^i$, из набора за предыдущий момент времени. Далее алгоритм начинает работу со случайной частицы, например, с номером m (выбор случайной частицы из набора существующих - например, функцией "$randi(N)$", которая генерирует случайное целое число в диапазоне от 1 до N):

1. Создается некая переменная β=0;
2. Присваиваем β случайное число от нуля до удвоенного максимального веса (т.е. $2\max_n \bar{\omega}_n^i$);
3. Сравниваем значение β со значением веса текущей частицы, $\bar{\omega}_m^i$:
   a. Если значение β больше веса частицы $\bar{\omega}_m^i$, то из значения β вычитается вес частицы $\bar{\omega}_m^i$ и далее переходим к рассмотрению следующей частицы в массиве (путем простого перебора по порядку);
   b. Как только β становится меньше веса текущей частицы, копия текущей частицы добавляется в новый массив частиц и происходит возврат к п.2 данного алгоритма, где значение β снова инициализируется случайным числом от нуля до удвоенного максимального веса.

Таким образом, каждая частица переходит в новый массив с вероятностью, равной её весу, и может быть добавлена в новый массив несколько раз. После отсева снова необходимо провести нормализацию веса частиц.

Блок-схема работы алгоритма на шаге 3 приведена в Приложении Д.

Выход фильтра: Расчет координат пользователя на выходе.

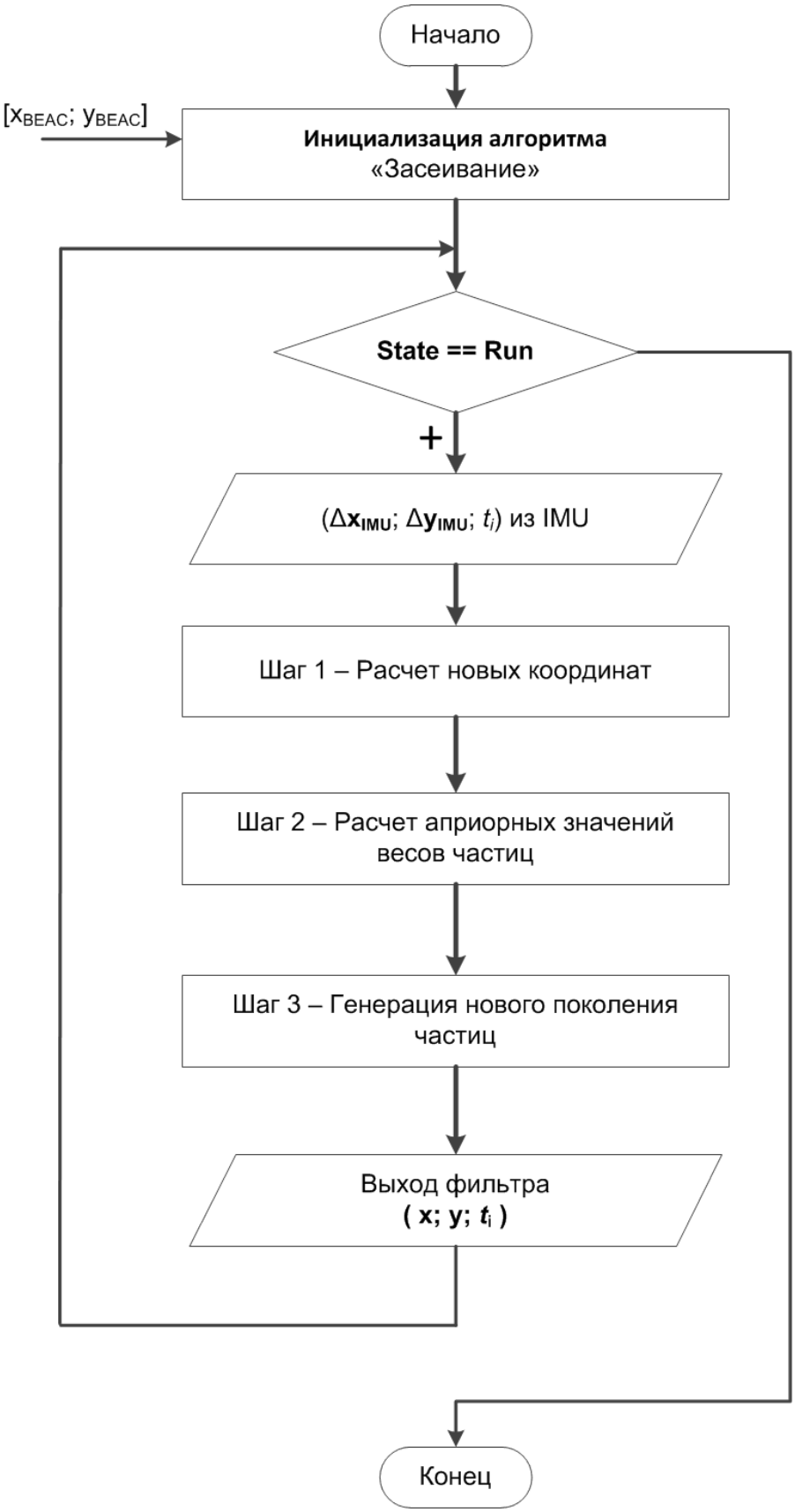
Расчет координат на выходе фильтра для заданного момента времени происходит путем расчета усредненного значения по всем частицам нового поколения по каждой координате по отдельности:

$$x_{OUT}(t_i) = \sum_{p=1}^{Np} newParticles(p).x, \qquad y_{OUT}(t_i) = \sum_{p=1}^{Np} newParticles(p).y$$

Далее алгоритм повторяется с **Шага 1**.

Блок-схема работы данного этапа алгоритма приведена в Приложении Е.

Приложение А – Общая блок-схема работы алгоритма



Приложение Б – Блок-схема работы алгоритма при Инициализации

Инициализация алгоритма
«Засеивание»

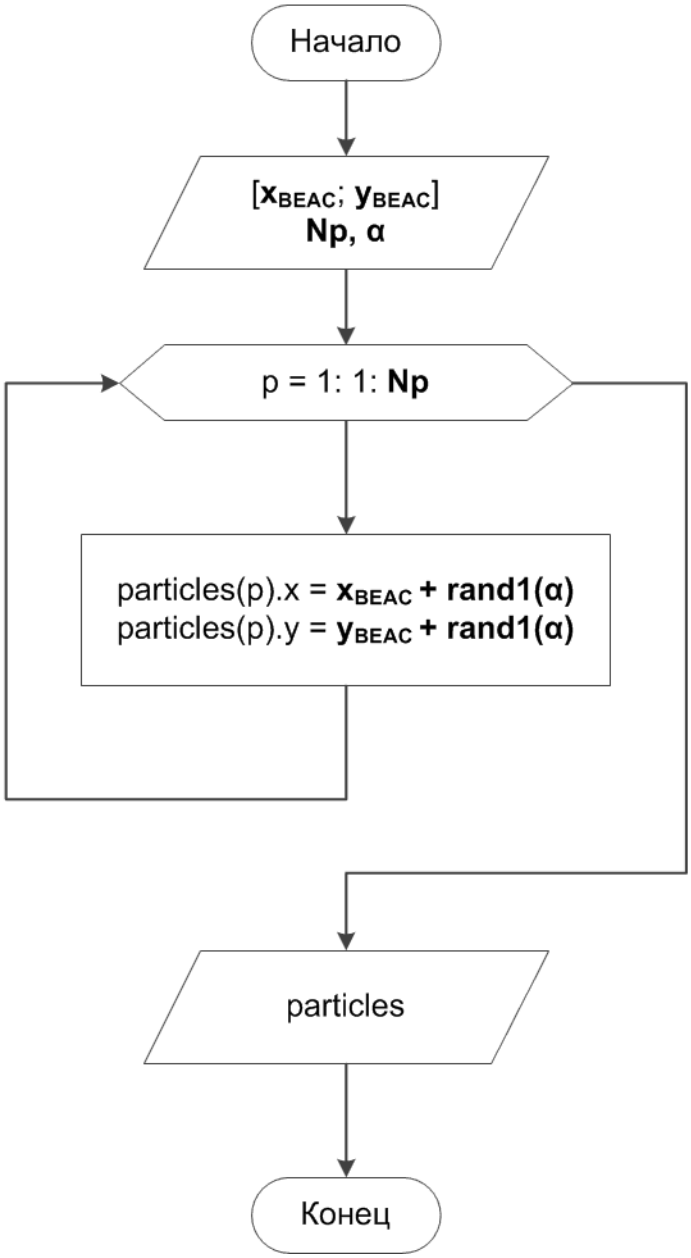
$[x_{BEAC}; y_{BEAC}]$ – координаты положения пользователя, определенные при помощи Beacons;

Np – количество частиц

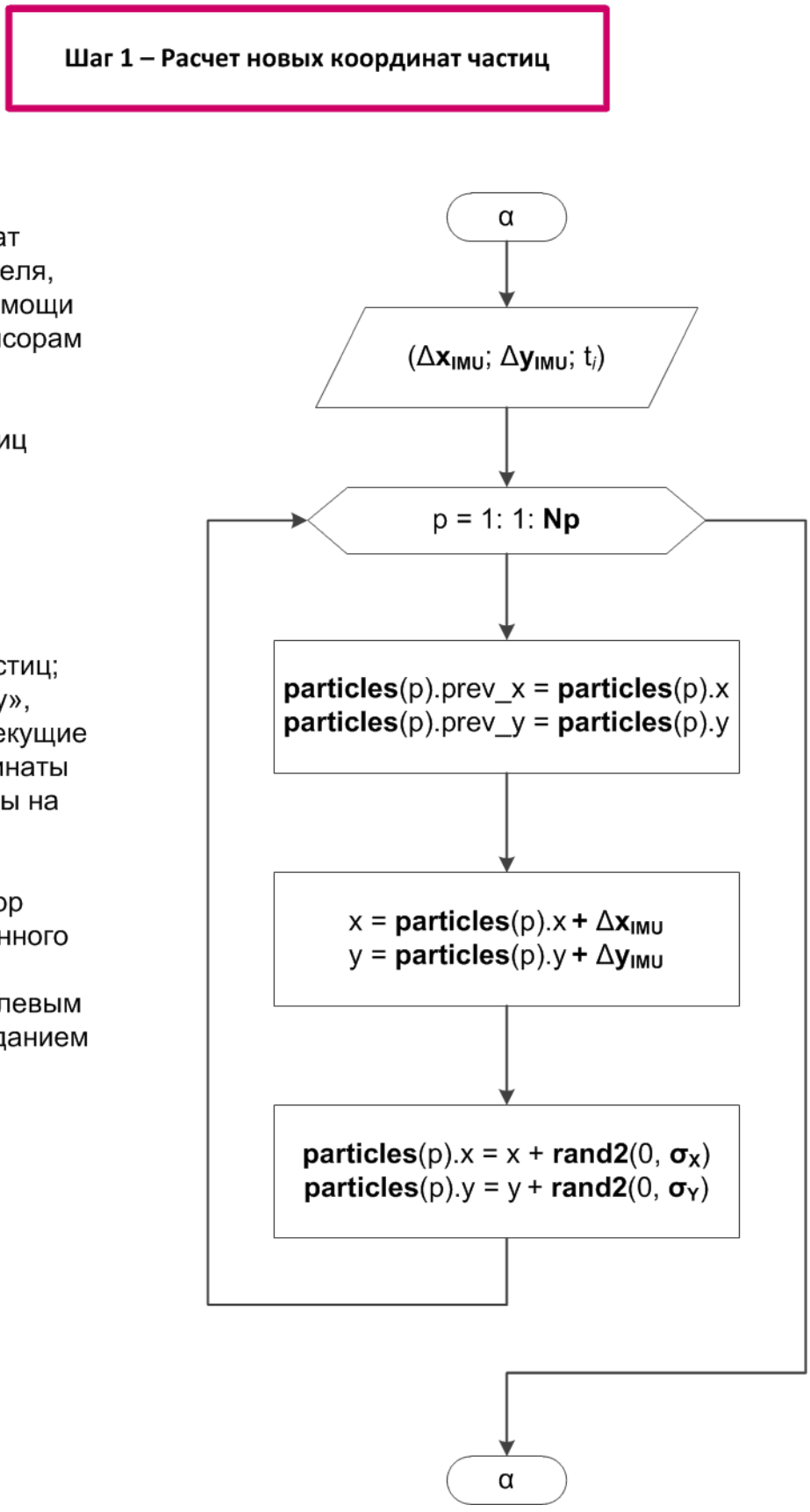
α – точность позиционирования по Beacons

particles – массив частиц; содержит поля «x», «y», «prev_x», «prev_y»- текущие и предыдущие координаты расположения частицы на карте;

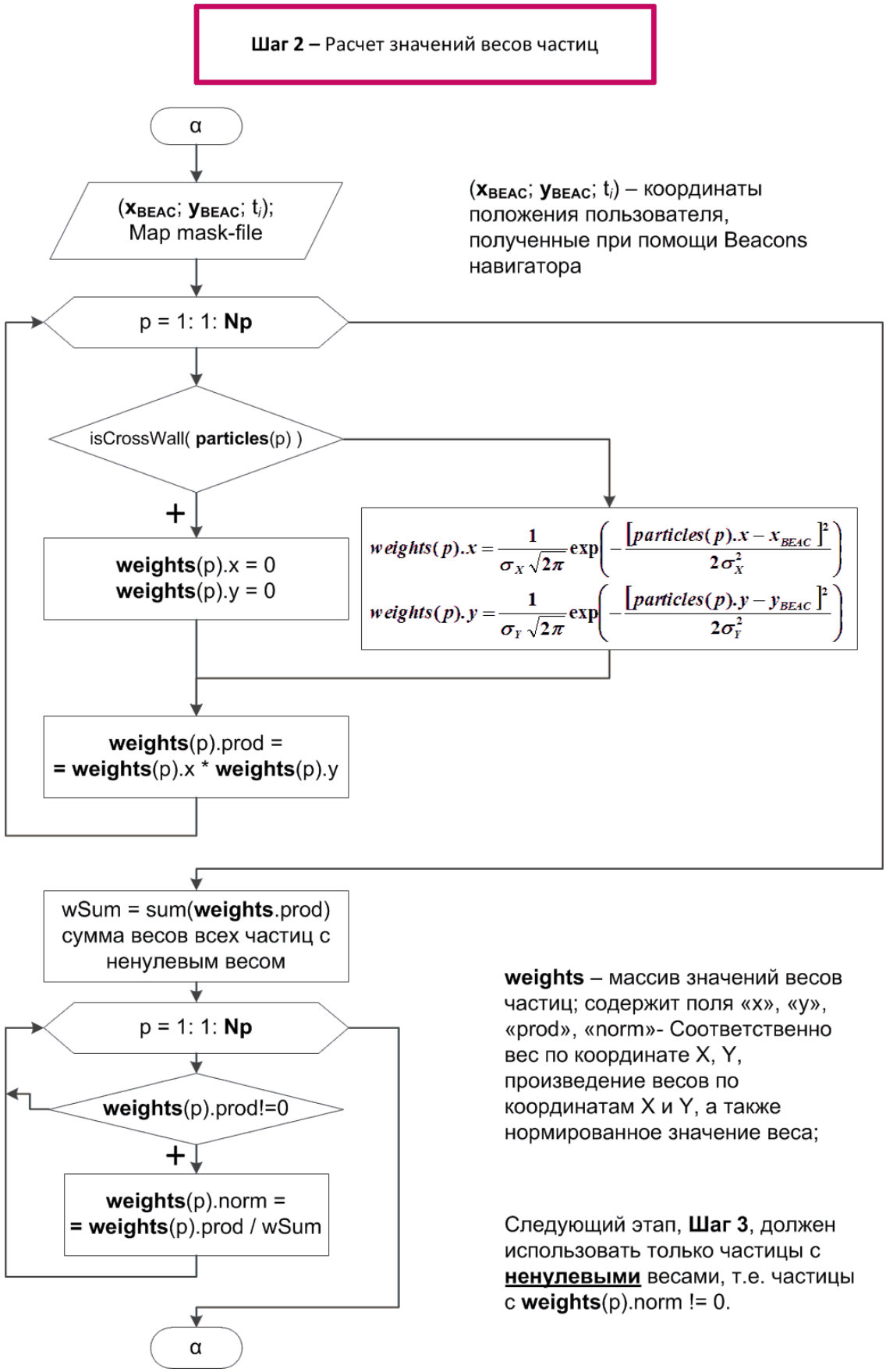
rand1(A) – генератор случайного вещественного числа с равномерным распределением в диапазоне [-α; α].



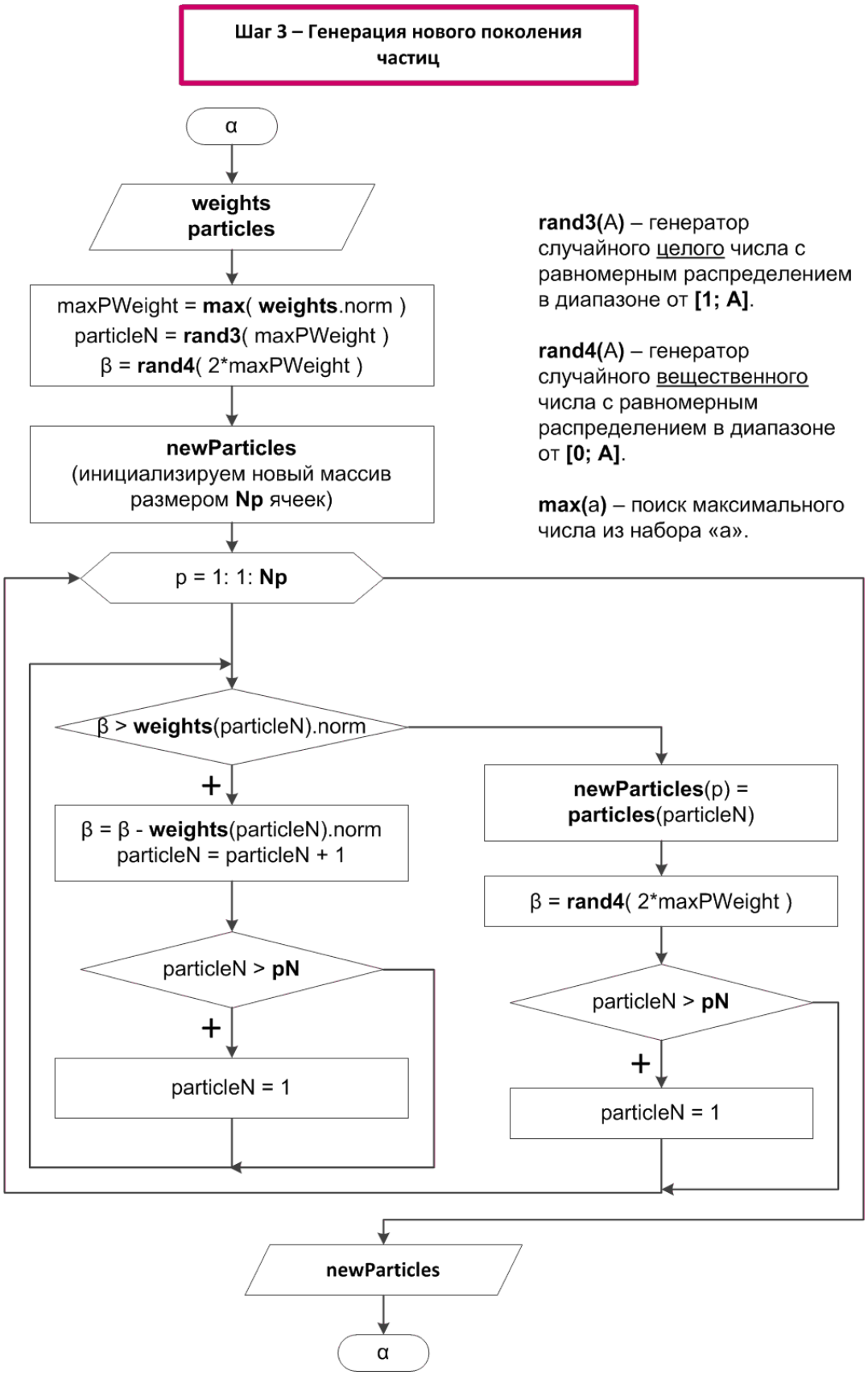
Приложение В – Блок-схема работы алгоритма на Шаге 1



Приложение Г – Блок-схема работы алгоритма на Шаге 2



Приложение Д – Блок-схема работы алгоритма на Шаге 3



Приложение Е – Блок-схема работы алгоритма в конце итерации