Devs based modeling and simulation of agricultural machinery movement

Article *in* Computers and Electronics in Agriculture · October 2020

DOI: 10.1016/j.compag.2020.105669

Computers and Electronics in Agriculture 177 (2020) 105669

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Devs based modeling and simulation of agricultural machinery movement

Ange-Lionel Toba[*], L. Michael Griffel, Damon S. Hartley

Idaho National Laboratory, 2525 Fremont Ave, Idaho Falls, ID 83415, United States

ARTICLE INFO

Keywords:
Field efficiency
Agriculture
Modeling and simulation
DEVS
Discrete event simulation
Bioenergy

ABSTRACT

Field operations planning has become critical for the operational efficiency of agricultural activities, in terms of time and cost. To address this type of planning problem, we (1) developed a modeling capability built using a Discrete Event Simulation (DES) approach and object-oriented prgramming, and (2) propose a methodology extending field data to allow spatial operations on various geometric types. The simulation model developed using the Discrete Event Specification System (DEVS) formalism, captures the equipment characteristics and tracks the evolution of its movement across the field via state transition tables. The model is data *driven* and spatially *and temporally detailed*, which are key features to accurately estimate field efficiency. Our modeling approach offers more rigor in the component behavior definition and more flexibility enabling the handling of fields and equipment of various shapes and physical characteristics, respectively. The validity of the model was tested, with an error margin of 15%, by comparing its performance with empirical data collected from switchgrass harvesting equipment.

1. Introduction

Integrated Landscape Management (ILM) has emerged as a more appropriate framework for balancing competing demands and integrating policies for efficient land uses, as opposed to sectorial approaches (Reed et al., 2015). ILM implies the management of resources on a long-term basis to achieve multiple economic, social and environmental objectives at a landscape scale. Note that ILM also presents potential negative impacts on the operability of fields. According to Place (2009), the absence of secure land tenure creates uncertainty and heavily limits farmers' ability to make long-term land-management or sustainability plans. This may result in the decrease of land reserves for agricultural activity. Also, the increasing demand drives the need to expand agricultural area in order to obtain short-term production increases, which is often done at the expense of forest and nature areas (Macqueen 2013). In an agricultural sector that has become so heavily dependent on mechanization, farmers aim to reduce production costs while maintaining high product quality. Machinery management and operation costs, which are a critical portion of the production costs, have generated substantial efforts from researchers to help farmers make informed decisions. Several approaches and techniques have been developed to analyze in-field equipment movement and ultimately estimate field efficiency.

This study uses a time-dependent field efficiency (FE) measure, as standardized by the ASAE EP496.3 (ASAE, 2000). Field efficiency is defined as the ratio between the real and the theoretical productivity of a machine. In the case of harvesting equipment, efficiency is often expressed as the ratio between productive time and the total time in the field. For this study, FE is calculated using the following Eq. (1):

$$FE = \frac{T_{work}}{T_{work} + T_{breaks}} \tag{1}$$

where T_{work} is the time to complete the operation with no delays or disengagement, including time for both headland and in-field movements, and T_{breaks} (time disengaged during breaks) is the time spent when the equipment is disengaged due to turning. T_{breaks} is likely impacted by field area and geometry, which affect the time spent turning while disengaged, the number of turns required during the operation at headlands or for navigating around obstructions, and the distance traveled in the headland space.

Using statistical analysis, Luck et al. (2011), Oksanen (2013) and Griffel et al. (2019) evaluate correlations between surface area descriptors and operational efficiency. Path finding techniques for equipment, with headland turning algorithms (Xue and Grift 2011) have also been used, tracking movement gradually from start to goal while avoiding obstacles (Zafar and Mohanta 2018). Classical approach including Cell Decomposition Method (Šeda 2007), or Grid Based Method (Schütz et al., 2014) offers more facility for implementation, though may require more details regarding the navigational environment, and ultimately more precise sensors. Heuristic approaches such

* Corresponding author.
E-mail address: danhoangelionel.toba@inl.gov (A.-L. Toba).

https://doi.org/10.1016/j.compag.2020.105669
Received 13 March 2020; Received in revised form 12 July 2020; Accepted 25 July 2020
Available online 07 August 2020
0168-1699/ Published by Elsevier B.V.

as fuzzy techniques (Oriolo et al., 1998), ant colony optimization (Bakhtiari et al., 2013) or even a combination of ant colony, simulated annealing, and framed-quad tree in order to enhance the efficacy of path planning (Zhang et al., 2012) offer more laxity when it comes to data strictness and are formulated as optimization problems. In his study, Oksanen (2007) formulated the area coverage planning problem as an Optimal Control Problem, aiming to find the optimal trajectory. Adamchuk et al. (2011) developed an algorithm for traffic pattern processing, resulting in the production of a map showing data layers associated with metrics evaluating machinery performance (cost of operation, capacity, efficiency, etc.). Another algorithm presented by Backman et al. (2015) helps generate a smooth path for headland maneuvering, accounting for curvature, speed, maximum steering rate and the maximum acceleration.

One of the limiting factors for numerical optimization is the computational complexity, especially considering multi-objective optimization algorithms (Fang et al., 2005). Also, the algorithms often require expensive computational power and sometimes do not provide a solution within a reasonable clock time window. Another limitation lies in the need for precise data for testing, which tend to be scarce. Statistical approaches used appear to provide less strong relationships with different combinations of field shape factors.

One viable option is the use of simulation. Simulation helps analyze the behavior of the real system through experiments, reach some understanding, and further learning, especially when access to empirical data is limited, as is often the case with agricultural equipment data. Simulation provides a means to observe and perceive how the behavioral trends of the model unfold, not only within its domain of applicability via validation testing (Sargent 2010), but also outside of it. This perception would constitute inference as to the properties of the behavior of the real system, and help predict the performance of such a system for any set of operating conditions using information on the design of its constituent components (Kim et al., 2017).

Several simulation models have been developed and used to analyze complex agricultural operations. Benson et al. (2002), De Toro and Hansson (2004) and Berruto et al. (2007) used a Discrete Event Simulation (DES) model to simulate the machinery movement for harvest. Dyer and Desjardins (2003)'s simulation model analyzes the amount of greenhouse gas emissions attributed to agricultural machinery management. Bochtis et al. (2009) model simulates both un-controlled and controlled traffic farming (UCTF and CTF) operations. Wold et al. (2011) developed a MATLAB program to simulate grain harvest and residue collection operations during a single pass harvest to evaluate grain collection and field operations efficiency.

Despites all of these efforts, these simulation models only focus on factors of a single operation with low spatiotemporal resolution, and for certain, lack in detail of in-field machine activities during the execution phase, including geometrical attributes of the operations environment. To bridge these gaps, Hameed et al. (2012) proposed an object-oriented implementation of the mathematical formulation of discrete events regarding the motion of the machines developed by Bochtis et al. (2009). However, their modeling approach does not apply to concave fields, therefore limiting its applicability to fields of certain shapes.

The object-oriented approach helps simplify the system complexity and increase the flexibility of the simulation model. This approach warrants a more maintainable and less tightly coupled design, facilitating the development, utilization and maintenance of the model. DES has a long history of association with the object-oriented paradigm, providing the critical ability to study the dynamic behavior of models that are defined with object-oriented means (Zeigler 1991). In this study, a similar approach is adopted, with the use of a modeling formalism, which according to Zeigler et al. (2000), has sound mathematical foundation and rigorous semantics. That way, the model is built in a structured manner following clear guidelines to specify the behavior of each component, thus avoiding ambiguity and enabling validation in a logically consistent fashion.

Our work contributes to the literature, (1) at the modeling level by developing a modeling capability using a formal DES paradigm and object-oriented approach, and (2) at the field operations planning level, developing a methodology extending current empirical field data to allow spatial operations on various geometric types. In Section 2, a brief definition of the DEVS formalism is given. In Section 3, we present the model used for the simulation, and discuss its characteristics. In Section 4, the methodology to estimate FE is explained. Section 5 presents the results and a brief discussion regarding the bien-fondé of this approach. The conclusion ensues in Section 6.

2. Model DEVS formalism

DES is a modeling paradigm (Sokolowski and Banks 2010), capturing the operation of a system as a discrete sequence of events in time. Each event -what triggers changes in the state of the system- occurs at a particular instant (Robinson 2004). This resolution adheres to the architecture of the simulation system built for this analysis. The position of the equipment on the field is tracked at specific/discrete time instances. For example, given a speed and width of 1 m per second and 1 m respectively, the equipment would cover 1 square meter each second. Thus, after each second, there is a new amount of field to cover, which represents a change in state of the system.

DEVS describes the system component behavior via *atomic* models using state transition functions, and the system structure via *coupled* models specifying connections between input/output ports. Both model types have input/output ports, which allow information to pass between them. Changes in the system are triggered by state transitions, which correspond to events, and take place at discrete time units. Zeigler (1976) formally defines a system as a tuple $S = (T, X, Y, Q, q_0, \delta, \lambda)$, where T is the time base determining lifespan of a given state , X is the set of inputs, Y is the set of outputs, Q is the set of states, q_0 is the initial state, δ: $Q \times X \rightarrow Q$ is the transition function, and λ: $Q \rightarrow Y$ is the output function. See table 1 for more details. DEVS formalism is appropriate for event-driven models; that is, systems whose states change anytime an event takes place. The state trajectories or changes are produced by state transition functions δ_{ext} and δ_{int} that are activated by external or internal events, respectively (Zeigler 1976).

The function δ is represented by both δ_{ext}: $Q \times X \rightarrow Q$ and δ_{int}: $Q \rightarrow Q$, and the system further described as a tuple $S = (T, X, Y, Q, q_0, \delta_{ext},$

Table 1
System specification.

Tuple	Description	Definition
$S = (T, X, Y, Q, q_0, \delta, \lambda)$	Formal definition of a system S, as a 7-tuple	
λ	Output function	λ: Q → Y function defining how a state of the system generates an output event.
Q, q_0	Set of sequential states, with q_0 being the initial state.	
X	Set of input events.	
Y	Set of output event	
T	Time advance function.	T: Q → $R^{+}_{0, +\infty}$ function defining state lifespan, as a positive real number, including infinity
$\delta = \{\delta_{int}, \delta_{ext}\}$	Internal and external transition function.	δ_{ext}: Q × X → Q function defining how an input event changes a state of the system δ_{int}: Q → Q function defining how a state of the system changes internally.

δ_{int}, λ). δ_{int} (s) = s' means that δ_{int} enables the transition from state *s* to *s'*, triggered by an internal mechanism, and δ_{ext} ((s, *e*), x) = s' enables the transition from state *s* to *s'* triggered by external input *x* with *e* being the elapsed time (time since the last transition). T (s) = σ means state s has a lifespan of σ time units.

3. Model description

This model captures the activities of agricultural equipment and the impact of field geometry on equipment performance. The model considers the *equipment* model, *field* model and *farm* model. Each system component is built in a formal manner and tested individually, according to its specific functionality. The *equipment* model (*atomic*), which represents agricultural equipment, is only concerned with a given activity on a given field. The *field* model (*atomic*) is concerned with information handling and computations; it acts as a regulatory agent, tracking progress made by the *equipment* agents, updating information and ensuring the right information is dispatched to the right agent. The *farm* model (*coupled*) handles the connections between all atomic models, specifying which input is linked to which output, ensuring the models are formally assembled and satisfactory behavior of the overall model. This is the model simulated by the DEVS simulator.

Each of these models are built using Python v.2.7, using *NumPy* package, *Bokeh* and *Shapefile* libraries, and object-oriented Programming. These entities or agents are modeled as objects, with each of them categorized in a class, with methods that implement appropriate process behaviors. These process-implementing classes will inherit from an appropriate atomic and coupled DEVS classes in the simulation package library (Van Tendeloo and Vangheluwe 2018).

3.1. Simulation platform architecture

An overview of the simulation architecture is displayed in Fig. 1. The component repository is composed of *atomic* and *coupled* DEVS models described by classes. Instances of these different models are generated based on the availability of data. The first step is the *data query*. Data are retrieved from CSV files containing information regarding the attributes of the models to be created. For example, for the *equipment* DEVS atomic model to be created, information fed to the model generation includes the track speed, width and the turning speed. Once all necessary data are retrieved, models are generated. This is the second step. The DEVS model generator creates all models by parameterizing and interconnecting instances of these generic classes available in the repository. Because the *farm* model describes the connections between *equipment* and *field* models, these have to be built prior. Once created, the *farm* model can then be created and simulated as part of the third step. The results of the simulation can be observed in the fourth and final step.

3.2. Model implementation

3.2.1. Field model: State and event description

Figs. 2 and 3 display the specification and state diagram of the *Field* model, respectively. The initial state is *idle*, lasts for 0 simulation time. With the state and timeout defined, the next step is the definition of the next state. This is the role of the internal transition function δ_{int} which triggers the transition to state *send*.

Similar to the previous state, the timeout is also defined, with the lifespan set to 0 simulation time as well. However, there is an associated output triggered, via the output function λ. The output *infoToEquipment* -distance to cover by the equipment- is sent to the *equipment* DEVS model. This needs to happen before the internal transition can trigger the next state transition. The model subsequently transitions from state *send* to state *wait*. Unlike the other states, the state *wait* does not have a timeout, rather it lasts for infinity or $+\infty$. This indicates the need for an external input to initiate transition. Like δ_{int}, δ_{ext} is still dependent on the current state. The external transition function has access to two values: the elapsed time *e* (used to track the total simulation time) and the bag of input events X. The input from the *equipment* model, *reportFromEquipment* -distance covered- once received, allows state transition via $\delta_{ext.}$

If the updated remaining distance is greater or equal to 0 (more details will be given in Section 4.2.), the model state transitions from *wait* to *advance*. Otherwise, it transitions to state *send.* If the state is *send*, the remaining distance is therefore re-computed and re-sent to the *equipment* model, during the same simulation time. If the state is *advance*, the simulation time is incremented by 1 (lifespan of state *advance*) and δ_{int} is again initiated. The model state transitions from *advance* state to *send* state. The simulation then progresses.

At state *advance*, one condition is checked to stop the simulation, which is to verify that all track distances are covered. That is, anytime the model state is *advance*, this condition is tested. If there is no more

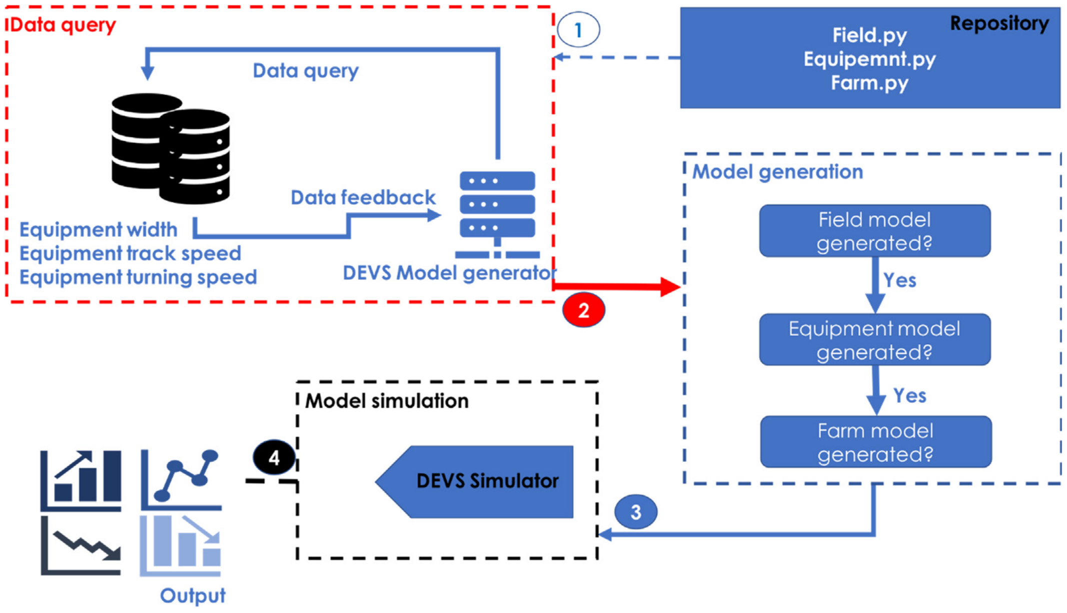

Fig. 1. Simulation Model architecture.

$$X = \{\mathit{reportFromEquipment}\}$$
$$Y = \{\mathit{infoToEquipment}\}$$
$$Q = \{\mathit{idle}, \mathit{wait}, \mathit{send}, \mathit{advance}\}$$
$$T\ (q \in Q) = \{T(\mathit{idle}) = T(\mathit{send}) = 0,\ T(\mathit{wait}) = +\infty,\ T(\mathit{advance}) = 1)\}$$
$$\delta_{int}\ (q \in Q) = \{\delta_{int}\ (\mathit{idle}) = \mathit{send},\ \delta_{int}\ (\mathit{send}) = \mathit{wait}\}$$
$$\delta_{ext}\ ((e, (q \in Q)), x \in X) = \{\delta_{ext}\ ((\mathit{wait}, e), \mathit{reportFromEquipment}) = \{\mathit{send}, \mathit{advance}\}\}$$
$$\lambda\ (q \in Q) = \{\ \lambda(\mathit{send}) = \mathit{infoToEquipment}, \lambda(\mathit{idle}) = \lambda(\mathit{advance}) = \lambda(\mathit{wait}) = \{\}\}$$

Fig. 2. *Field* model specification.

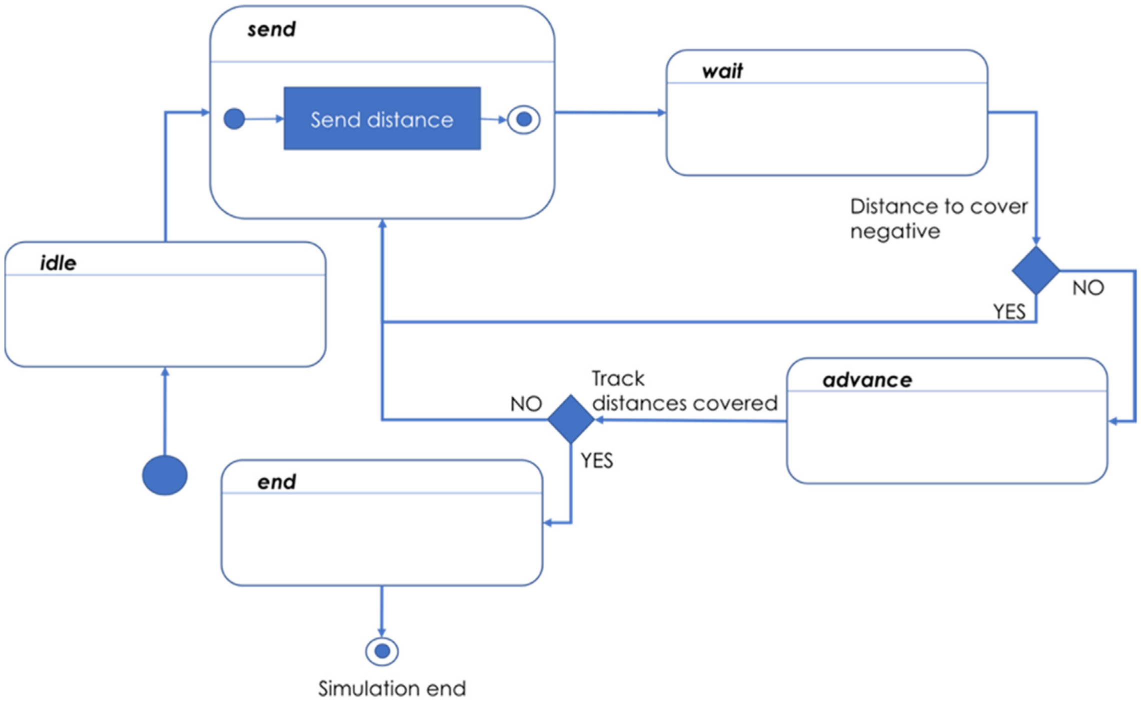

Fig. 3. State diagram of the *Field* model.

track distance, the simulation stops, and the transition function is not initiated.

3.2.2. Equipment model: State and event description

Figs. 4 and 5 display the specification and state diagram of the *equipment* model, respectively. The initial state is *idle*, which lasts for 0 simulation time. The internal transition function δ_{int} is initiated, which triggers the transition to state *wait*.

Similar to the *Field* model, the *wait* state lasts for infinity or $+\infty$, expecting an input to trigger a transition. The input is from the *Field* model -*infoFromField* (remaining distance to cover)-. Once received, δ_{ext} is initiated, and the model state evolves to *report* state. Here too, there is an associated output triggered, via the output function λ. The output *reportToField* -distance covered by the agricultural machine over a simulation time- is sent to the *Field* DEVS model. This needs to happen before the internal transition can trigger the next state transition. The model transitions from state *report* to state *wait*. The cycle repeats.

3.2.3. Farm model: State and event description

This model describes the structure of the system, and therefore is not defined by state transition, output or time advance functions. Instead, it specifies the links between input and output of atomic models. All messages are sent/received via specific input/output ports, which need to be defined.

$$X = \{\mathit{infoFromField}\}$$
$$Y = \{\mathit{reportToField}\}$$
$$Q = \{\mathit{idle}, \mathit{wait}, \mathit{report}\}$$
$$T\ (q \in Q) = \{T(\mathit{idle}) = 0,\ T(\mathit{wait}) = +\infty,\ T(\mathit{report}) = 0)\}$$
$$\delta_{int}\ (q \in Q) = \{\delta_{int}\ (\mathit{idle}) = \mathit{wait},\ \delta_{int}\ (\mathit{report}) = \mathit{wait}\}$$
$$\delta_{ext}\ ((e, (q \in Q)), x \in X) = \{\delta_{ext}\ ((\mathit{wait}, e), \mathit{infoFromField}) = \mathit{report}\}$$
$$\lambda\ (q \in Q) = \{\ \lambda(\mathit{send}) = \mathit{reportToField}, \lambda(\mathit{idle}) = \lambda(\mathit{advance}) = \lambda(\mathit{wait}) = \{\}\}$$

Fig. 4. *Equipment* model specification.

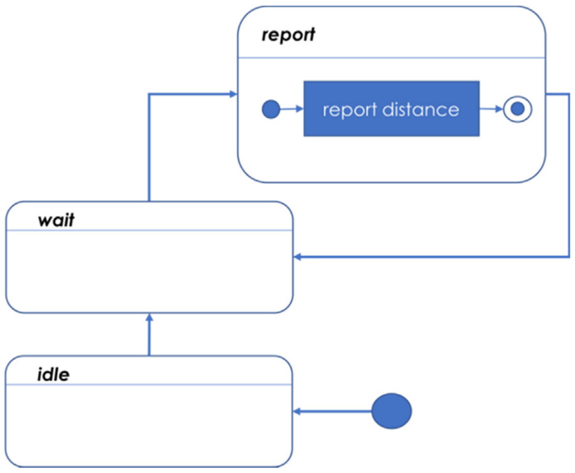

Fig. 5. State diagram of the *Equipment* model.

4. Methodology and simulation mechanism

4.1. Methodology steps

The methodology presented is composed of steps taken to estimate field efficiency. We used geo-referenced data of switchgrass mowing coverage that were collected at the end of the 2017 growing season in the eastern United States by FDC Enterprises Inc. (New Albany, Ohio). Visual examination of the equipment movement shows operators making several headland passes around the contours of a field, before transitioning to parallel back-and-forth or track passes to cover the interior section (Griffel et al., 2019). This pattern is what guided the step selection in our methodology.

Step 1 – Read and prepare geo-referenced data

All field boundaries were digitized using QGIS geographic information system (GIS) software (QGIS Development Team 2018) based on the harvest coverage data, collected at an approximate 1-second temporal resolution at the end of the 2016 growing season as described by Griffel et al. (2019). Fig. 6 shows what the coverage data looks like on a given field.

This process was utilized to generate 36 field boundaries, without obstacles, that were saved as polygon shapefiles. These files store shape and coordinate information (longitude and latitude) of points forming the polygon. These data are read by the *field* model, using the *GeoPandas* Python library, which combines the capabilities of *Pandas, Fiona* and *Shapely* to provide geospatial operations in Pandas and handle multiple geometries. For a given polygon shapefile, the model records all the vertices and computes the length of the edge between each 2 consecutive vertices. All lengths are summed up. A new polygon shapefile is generated, with edge length between each 2 consecutive vertices computed, and all lengths summed up again. This operation is repeated for as many headlands passes as desired. These constitute steps in task 1 performed by the *field* model. Fig. 7 shows the flow diagram code used to perform this task. The new polygon shapefiles are generated based on the width of the machine and the number of passes. Fig. 8 shows new field generated after each headland pass.

Step 2 – Find the longest distance in order to minimize the number of turnarounds.

This operation is also performed by the *field* model. In this step, the minimum bounding rectangle contains the field geometry. This rectangle is an envelope, with edges not necessarily parallel. Once found, the longest edge of the rectangle is determined. This is the *field* model

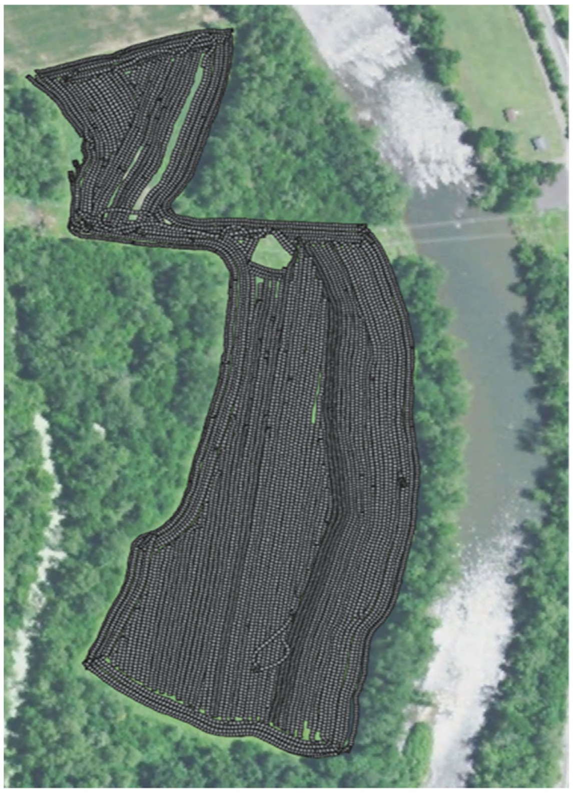
Fig. 6. Example of field with switchgrass mowing coverage data.

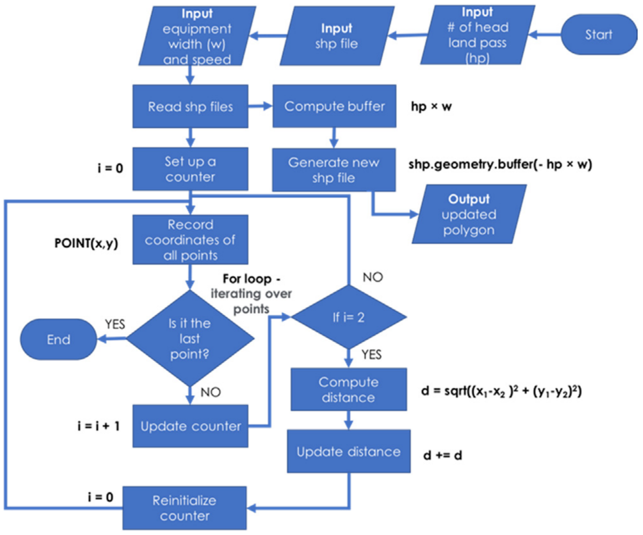

Fig. 7. Task 1 flow diagram Python code.

2nd task, displayed in Fig. 9. The reason for this approach is that visual analysis of the empirical data suggests equipment operators typically orientate their paths to approximate the longest edge of the field to minimize turn arounds. Minimal turnaround time would ultimately mean minimal break time. It is assumed that turnaround time is a break time, time during which the equipment is not productive. Fig. 10 shows the envelope generated, containing the field.

Step 3 – Find the number of turnarounds.

Lines representing equipment pathways and parallel to the longest edge of the envelope found in the previous step are established. For that, we use the equation of that edge, and deduce equations of the

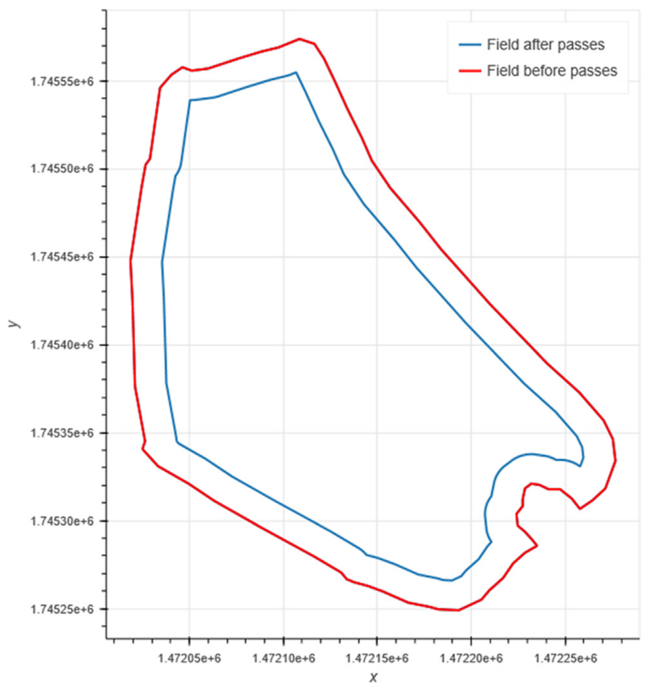

Fig. 8. Visual representation of shape files generated.

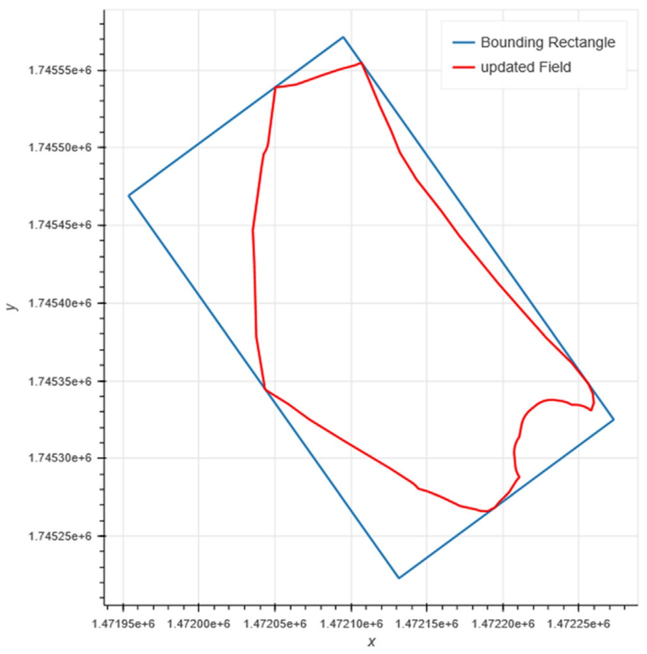

Fig. 10. Visual representation of envelope generated.

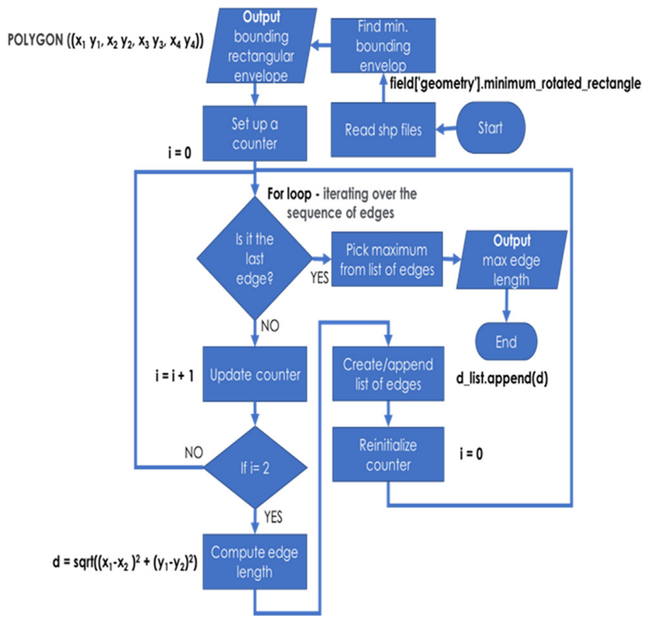
POLYGON ((x1 y1, x2 y2, x3 y3, x4 y4))
Output bounding rectangular envelope
Find min. bounding envelop
field['geometry'].minimum_rotated_rectangle
Set up a counter
Read shp files
Start
i = 0
For loop - iterating over the sequence of edges
Is it the last edge?
YES
Pick maximum from list of edges
Output max edge length
NO
End
i = i + 1
Update counter
Create/append list of edges
d_list.append(d)
NO
If i = 2
Reinitialize counter
i = 0
YES
d = sqrt((x1-x2)² + (y1-y2)²)
Compute edge length

Fig. 9. Task 2 flow diagram Python code.

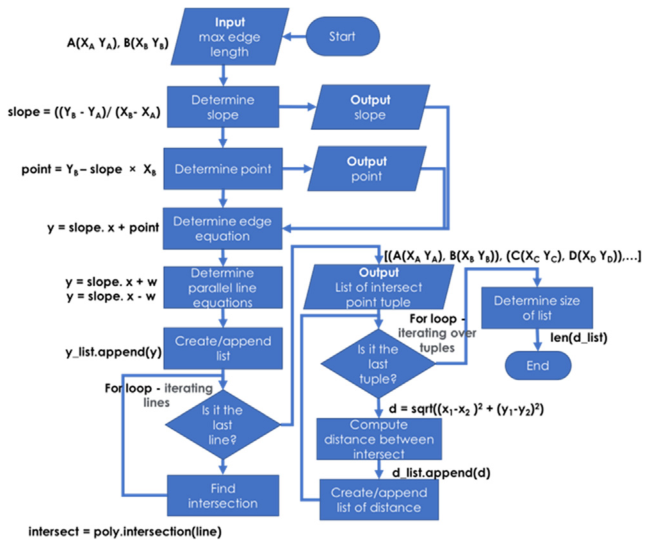

Fig. 11. Task 3 flow diagram Python code.

parallel lines. In drawing these lines, we record intersecting points with the polygon boundaries. Each intersecting point represents a point at which the equipment turns around. The distance between intersect on the same line (tuple of points) is computed, as it represents the track distances. The number of intersect points, which correspond to half the number of turnarounds of the equipment, is then recorded. This is task 3 of the field model (Fig. 11). Fig. 12 shows the parallel lines drawn.

Step 4 – Compute FE

In this step, FE is simply computed, using equation (1) presented in the introduction. Time is obtained when dividing distance by speed. In this study, T_{work} equals distance covered during headland and in-field movements (computed in tasks 2 and 3) divided by the speed of the agriculture equipment. T_{breaks} equals distance covered during turnarounds divided by the speed of the agriculture equipment. The speed is assumed the same during turnarounds, headland and in-field movements.

4.2. Simulation mechanism

The simulation mechanism is represented in Fig. 13. The time step used here depends on the unit of the speed of the equipment. If it is in meter/sec, the time step is 1 s. At the beginning of the simulation, data are retrieved from databases containing width of the *equipment* agent, as well as the number of headland passes, to cover the periphery of the field and shapefile documents. The *field* agent performs task 1 while at state *idle*. Once at state *send*, it sends a DEVS output message to the *equipment* agent with the total distance to cover all headland passes (displayed as 1 on the figure), before switching to state *wait*. The *equipment* model receives the message while at state *wait*. A transition is initiated to state *report*, triggering a response back to the *field* model

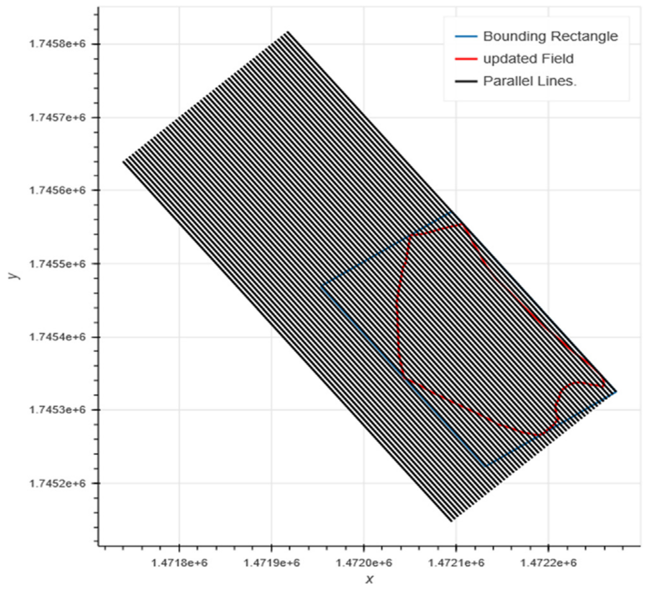

Fig. 12. Visual representation of parallel lines.

(message 2). This response is the distance covered in a simulation time unit, based on its speed. Its state is then transitioned back to *wait*. The *field* model receives the response, updates the distance to cover. If the updated distance to cover or remaining distance is negative, it transitions to state *send* and re-sends the distance to the *equipment* model, during the same simulation time unit. That means the equipment covered more distance than needed. This situation happens when the remaining distance is shorter than the distance covered conditioned by the *equipment* constant speed, within a simulation time. The equipment model readjusts itself and returns the appropriate value. If the distance is positive, the *field* model transitions to *advance*, then to *send*. This exchange (message 1 – message 2) continues until the whole distance for headland passes is covered -remaining distance for headland passes equals 0-. The time to cover these passes is recorded.

At that point, it is time to initiate track passes. The *field* model performs tasks 2 and 3 to obtain the distance for track passes, as well as the number of turnarounds and the time for turning, while at state *advance*. The internal transition function is initiated, and the state is transitioned to *send*. The *field* then re-sends messages to the *equipment*

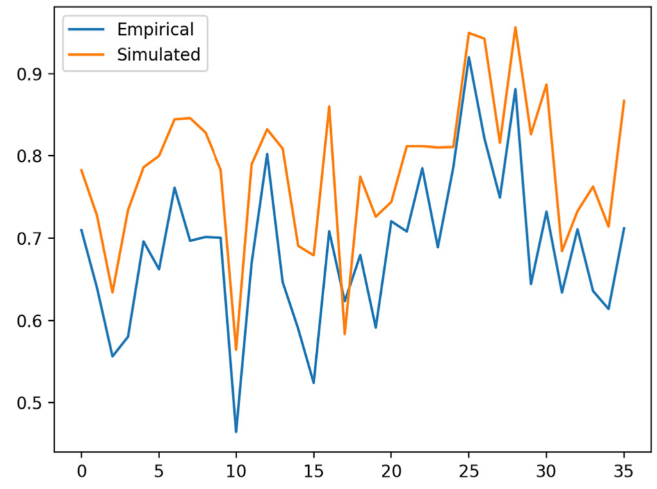

Fig. 14. Results comparison between empirical and simulation.

with information regarding track passes to cover the inside of the field (message 3). In message 4, output DEVS are sent back to the *field* agent, specifying the distance covered in a simulation time unit. This exchange follows the same state transitions and conditions as the previous one (message 1 – message 2). The time to cover these passes is also recorded. Once the remaining distance for track passes equals 0, FE can be computed. The *field* model then transitions from state *advance* to *end*.

5. Simulation result and discussion

The functionality and validity of the simulation model is demonstrated by applying it to a variety of fields, with different shapes, different equipment with different speed and widths, as well as different number of headland passes. The simulation results are compared with the empirical field efficiency used and analyzed by Griffel et al. (2019). Harvesting FE for the 36 fields of switchgrass cutting operations ranges from 0.4645 to 0.9200. The simulation model provides FE results ranging from 0.5646 to 0.9562, with an average of 14.88% error. Fig. 14 displays the trends followed by FE values, and Fig. 15, the error on FE estimation for each field.

As any modeling exercise, capturing all relevant detail of the real system is challenging. In our case, the *equipment* agent in fact describes the behavior of drivers of the equipment in real life. This model does

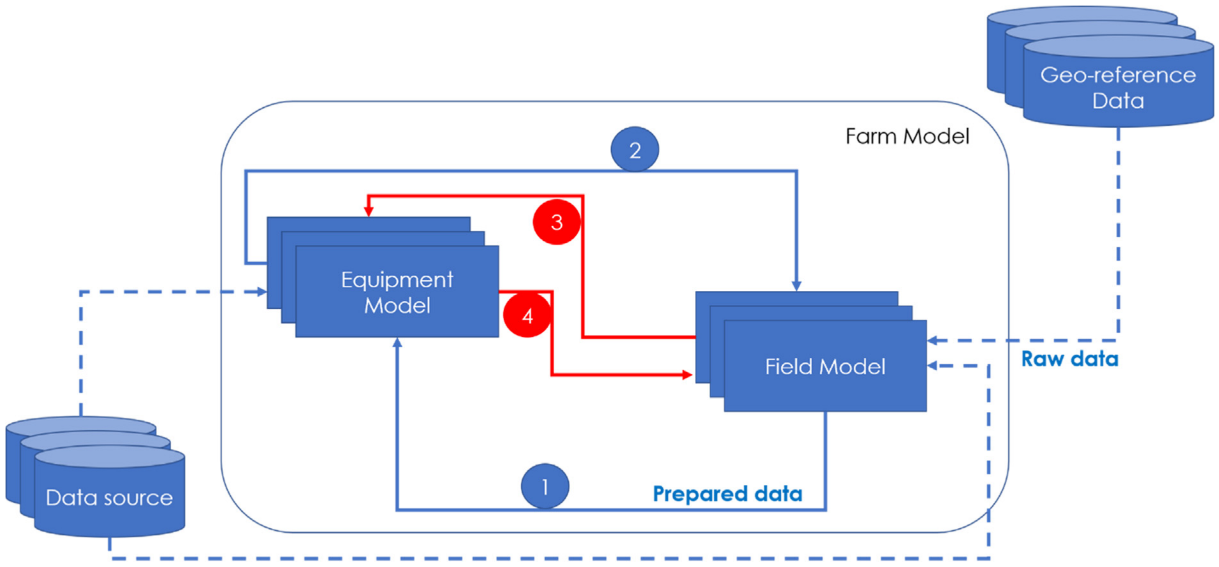

Fig. 13. Modeling conceptualization of operations.

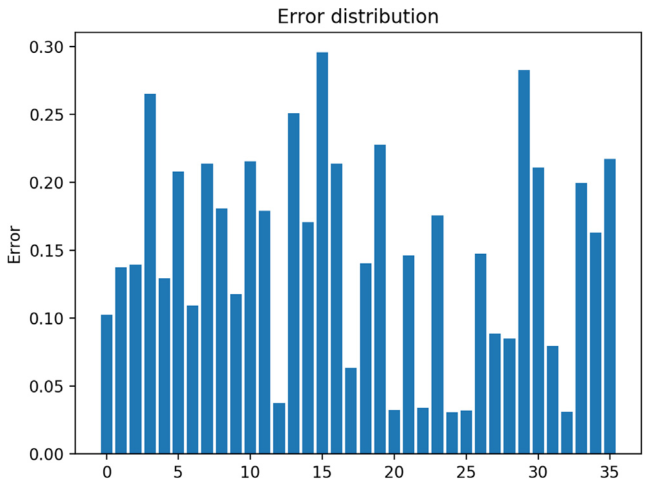

Fig. 15. Error distribution between simulated and empirical results.

not account for their choices in terms of breaks (due to fatigue, restrooms, food, etc.) or in terms of arbitrary direction while maneuvering the equipment. Mechanical malfunction is not accounted for either. Another factor not accounted for is the turnaround speed, which we assume to be equal to the linear (track) speed. Realistically, this speed should be lower. We posit that these details affect the accuracy of our results.

This study showed how agricultural systems may be advantageously mapped into DEVS representations. Discrete event representations afford thus a more efficient simulation. The challenge is to be able to match states to prerogatives or actions performed by components in the system. Actions associated with states and transitions govern the behavior of the modeled entity. Each state and associated actions represent a distinct behavior phase that the entity display. In this case, it was also important to first discretize time, that is, determine the points in time at which events would occur and states would transition. This choice was bound by the speed unit of the equipment. If the speed is defined as the amount of ground or distance covered per amount of time, this time unit would constitute for us that point of time.

One requirement for the simulation model needed was to be data driven, which offers more room for extensibility. In that sense, in addition to specifying and analyzing a specific field and equipment, a whole class of similar systems (though with different equipment speed, width, field shapes, tasks, etc.), given appropriate instantiation and dimensioning, could also be analyzed. This level of detail brings more credibility to the results.

Through the use of simulation, the variations of the state of the model's variables over time are examined and ultimately the performance of the system is evaluated, under different configurations. Simulation is therefore a means to observe and perceive how the behavioral trends of the model unfold. This perception would constitute inference as to the properties of the behavior of the real system. The objective in developing this capability is to provide farmers opportunities to estimate machinery performance without conducting time consuming field experiments, and subsequently make better managerial or technical decisions.

6. Conclusion

The planning of field operations is an essential task for the operational efficiency in terms of time and cost. Field efficiency is the efficiency measure chosen for this study. This paper documents efforts in developing a modeling capability, using Discrete Event Simulation and DEVS formalism, and a methodology extending field data to allow spatial operations on geometric types, to simulate movements of agricultural machinery and estimate FE. The model is data driven, offering flexibility to users to input a wide variety of parameters, spatially and temporally detailed offering more precision. In using a modeling formalism and object-oriented technology, only one semantic interpretation is allowed, in order to facilitate the development and utilization of the model. It warrants a more maintainable and less tightly coupled design.

Empirical data were used to test the validity of the model. The model offers a 15% error margin, compared to data and FE analytically computed from over 30 fields of different shape and size. The results of this study prove the accuracy of the model, and we posit, position this modeling capability as a credible tool to assess field efficiency.

There is still room for improvement in this study. Given that the simulation model presented here is structured to account for gaps and overlaps, potential future work could explore this avenue. Another aspect which was not included in our study is fields with obstacles. The model applies only to free-of-obstacles fields. Given that the understanding of how field shapes affect harvesting time in each field can help optimize the field design for higher efficiency, extending the model to fields with obstacles would be of great importance.

CRediT authorship contribution statement

Ange Lionel Toba: Conceptualization, Methodology, Software, Validation, Visualization. **L. Michael Griffel:** Supervision, Software, Conceptualization, Data curation. **Damon S. Hartley:** Conceptualization, Investigation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the US Department of Energy's Office of Energy Efficiency and renewable Energy, Bioenergy Technologies Office, under DOE Idaho Operations Office Contract DE-AC07-05ID14517. Accordingly, the US Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for US Government purposes. The authors thank the Antares Group and FDC Enterprises for their generous sponsoring support through a cooperative agreement with the U.S. Department of Energy titled: "Enabling Sustainable Landscape Design for Continual Improvement of Operating Bioenergy Supply Systems" (Award Number EE0007088).

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.compag.2020.105669.

References

Adamchuk, V.I., et al., 2011. Spatial variability of field machinery use and efficiency. GIS App. Agric. 2, 135–146.
ASAE (2000). ASAE standards 2000: standards, engineering practices, data, Inside New York.
Backman, J., et al., 2015. Smooth turning path generation for agricultural vehicles in headlands. Biosyst. Eng. 139, 76–86.
Bakhtiari, A., et al., 2013. Operations planning for agricultural harvesters using ant colony optimization. Spanish J. Agric. Res. 11 (3), 652–660.
Benson, E., et al., 2002. Development of an in-field grain handling simulation in ARENA. 2002 ASAE Annual Meeting. American Society of Agricultural and Biological Engineers.
Berruto, R., et al., 2007. Modeling of grain harvesting: interaction between working pattern and field bin locations. In: XXXII CIOSTA-CIGR V Labour and Machinery Management for a Profitable Agriculture and Forestry, Dept. of Machinery and Production Systems.

Bochtis, D., et al., 2009. Modelling of material handling operations using controlled traffic. Biosyst. Eng. 103 (4), 397–408.
De Toro, A., Hansson, P.-A., 2004. Analysis of field machinery performance based on daily soil workability status using discrete event simulation or on average workday probability. Agric. Syst. 79 (1), 109–129.
Dyer, J., Desjardins, R., 2003. Simulated farm fieldwork, energy consumption and related greenhouse gas emissions in Canada. Biosyst. Eng. 85 (4), 503–513.
Fang, H., et al., 2005. A comparative study of metamodeling methods for multiobjective crashworthiness optimization. Comput. Struct. 83 (25–26), 2121–2136.
Griffel, L.M., et al., 2019. Agricultural field shape descriptors as predictors of field efficiency for perennial grass harvesting: An empirical proof. Comput. Electron. Agric. 105088.
Hameed, I., et al., 2012. An object-oriented model for simulating agricultural in-field machinery activities. Comput. Electron. Agric. 81, 24–32.
Kim, B.S., et al., 2017. Data modeling versus simulation modeling in the big data era: Case study of a greenhouse control system. Simulation 93 (7), 579–594.
Luck, J., et al., 2011. A Case Study to Evaluate Field Shape Factors for Estimating Overlap Errors with Manual and Automatic Section Control. Trans. ASABE 54 (4), 1237–1243.
Macqueen, D., 2013. Landscapes for public goods: multi-functional mosaics are fairer by far. International Institute for Environment and Development.
Oksanen, T., 2007. Path planning algorithms for agricultural field machines. Helsinki University of Technology.
Oksanen, T., 2013. Shape-describing indices for agricultural field plots and their relationship to operational efficiency. Comput. Electron. Agric. 98, 252–259.
Oriolo, G., et al., 1998. Real-time map building and navigation for autonomous robots in unknown environments. IEEE Trans. Syst, Man, and Cybernetics, Part B (Cybernetics) 28 (3), 316–333.
Place, F., 2009. Land tenure and agricultural productivity in Africa: a comparative analysis of the economics literature and recent policy strategies and reforms. World Dev. 37 (8), 1326–1336.
QGIS Development Team. 2018. QGIS Geographic Information System, Open Source Geospatial Foundation Project.
Reed, J., et al., 2015. What are 'Integrated Landscape Approaches' and how effectively have they been implemented in the tropics: a systematic map protocol. Environ. Evidence 4 (1), 2.
Robinson, S., 2004. "Simulation: The Practice of Model Development and Use. Inglaterra: John Willey and Sons. Inc., Cap 1 (2), 5.
Sargent, R.G., 2010. Verification and validation of simulation models. Proceedings of the 2010 Winter Simulation Conference. IEEE.
Schütz, M., et al., 2014. Occupancy grid map-based extended object tracking. 2014 IEEE Intelligent Vehicles Symposium Proceedings. IEEE.
Šeda, M., 2007. Roadmap methods vs. cell decomposition in robot motion planning. Proceedings of the 6th WSEAS international conference on signal processing, robotics and automation. World Scientific and Engineering Academy and Society (WSEAS).
Sokolowski, J.A., Banks, C.M., 2010. Modeling and simulation fundamentals: theoretical underpinnings and practical domains. John Wiley & Sons.
Van Tendeloo, Y., Vangheluwe, H. 2018. Introduction to parallel DEVS modelling and simulation. In: Proceedings of the Model-driven Approaches for Simulation Engineering Symposium, Society for Computer Simulation International.
Wold, M.T., et al., 2011. Modeling the In-Field Logistics of Single Pass Crop Harvest and Residue Collection. 2011 Louisville, Kentucky, August 7-10, 2011. ASABE, St. Joseph, MI.
Xue, J.L., Grift, T.E. 2011. Agricultural robot turning in the headland of corn fields. Applied Mechanics and Materials, Trans Tech Publ.
Zafar, M.N., Mohanta, J., 2018. Methodology for Path Planning and Optimization of Mobile Robots: A Review. Procedia Comput. Sci. 133, 141–152.
Zeigler, B.P., 1976. Theory of modelling and simulation. Wiley, New York, NY.
Zeigler, B.P., 1991. Object-oriented modeling and discrete-event simulation. Adv. Comput., Elsevier 33, 67–114.
Zeigler, B.P., et al., 2000. Theory of modeling and simulation. Academic press.
Zhang, Q., et al., 2012. Path planning based quadtree representation for mobile robot using hybrid-simulated annealing and ant colony optimization algorithm. Proceedings of the 10th World Congress on Intelligent Control and Automation. IEEE.