From: AAAI Technical Report WS-97-08. Compilation copyright © 1997, AAAI (www.aaai.org). All rights reserved.

Spatial Aggregation: Modeling and controlling physical fields

Christopher Bailey-Kellogg **Feng Zhao**
Department of Computer and Information Science
The Ohio State University
2015 Neil Avenue
Columbus, OH 43210 U.S.A.
{kellogg,fz}@cis.ohio-state.edu

Abstract

Many important physical phenomena, such as temperature distribution, air flow, and acoustic waves, are described as continuous, distributed parameter fields. Controlling and optimizing these physical processes and systems are common design tasks in many scientific and engineering domains. However, the challenges are multifold: distributed fields are conceptually harder to reason about than lumped parameter models; computational methods are prohibitively expensive for complex spatial domains; the underlying physics imposes severe constraints on observability and controllability.

This paper develops an ontological abstraction and an aggregation-disaggregation mechanism, in a framework collectively known as **spatial aggregation** (SA), for reasoning about and synthesizing distributed control schemes for physical fields. The ontological abstraction models physical fields as networks of spatial objects. The aggregation-disaggregation mechanism employs a set of data types and generic operators to find a feasible control structure, specifying control placement and associated actions that satisfy given constraints. SA abstracts common computational patterns of control design and optimization in a small number of operators to support modular programming; it builds concise and articulable structural descriptions for physical fields. We illustrate the use of the SA ontological abstraction and operators in an example of regulating a thermal field in industrial heat treatment.

Keywords. Qualitative reasoning; Spatial reasoning; Ontologies; Decentralized control; Distributed AI; Programming languages.

Introduction

Continuous, distributed parameter fields are common physical phenomena: consider the temperature field in a building, the air flow around an airplane wing, or the noise from a copy machine. There are enormous practical benefits to reasoning about and controlling these physical processes and systems. For instance, the drag on an airplane can be reduced by analyzing and controlling the air flow around the wings. Temperature in a "smart" building can be regulated to maximize occupant comfort while minimizing energy consumption. Because of the rapid advance in micro-fabrication technology that can integrate and produce micro-electro-mechanical system (MEMS) devices on a massive scale, we are becoming increasingly reliant on large networks of sensors, actuators, and computational elements to augment our ability to interact with and control the physical environment (Berlin 1994).

However, there is enormous challenge in controlling and optimizing physical fields using networks of sensors and controllers. The difficulties arise from three sources. First, a distributed parameter field is conceptually harder to reason about and model than a lumped parameter system such as a circuit. Spatial topology, metric, material properties and physical laws all come into play, in addition to the combinatorial structures. The underlying physical processes might be nonlinear and defy analytic, closed form solution. Second, numerical methods developed for designing and controlling physical fields require solving large systems of equations and hence are prohibitively expensive for large, irregular geometric domains and highly non-uniform, nonlinear phenomena. Third, physical laws constrain the ability of spatially distributed, local agents to sense and affect the environment (Seidman 1996). Local sensors and control elements measure and interact with small neighborhoods around them. Macroscopic consequences are aggregated from local actions. Consequently, the design, programming, and coordination of distributed computational agents immersed in physical media require abstraction mechanisms, inference methods, and programming languages different from those for reasoning about and controlling centralized, lumped parameter models.

In this paper, we develop an ontological abstraction for physical fields and an aggregation-disaggregation mechanism for reasoning about and synthesizing distributed control schemes for physical fields. The ontological abstraction encodes neighborhood relations and equivalence classes and describes the structure and behavior of a field in multiple layers of spatial objects. The aggregation-disaggregation mechanism finds a feasible control design for a physical field specifying control placement and associated actions that meet given design objectives; it employs `spatial object`, `field`, and `ngraph` data types and generic operators such as **`aggregate`**, **`disaggregate`**, **`classify`**, **`interpolate`**, **`redescribe`**, and **`update`** that manipulate and transform objects. The ontological abstraction, generic operators, and aggregation-disaggregation mechanisms are collectively called **spatial aggregation** (SA); we have implemented the spatial aggregation language (SAL) to support modular programming in this framework. SA transforms higher-level control objectives into local control actions using divide-and-conquer, exploiting physical constraints of fields manifested as locality, continuity, and spatial and temporal scales. We illustrate the use of the SA ontological abstraction and operators in an example of temperature regulation of a thermal field from an industrial heat treatment problem. This work has significantly extended the previously developed SA framework for data interpretation (Yip & Zhao 1996; Bailey-Kellogg, Zhao, & Yip 1996) to address new problems arising from control and optimization of distributed parameter physical fields.

SA differs from existing numerical design and optimization methods in several important ways. First, numerical methods build a discretization for a physical field. In contrast, SA constructs an explicit structural description for the field from a numerical discretization, measurements, or other sources. This structural description can be abstracted and articulated to support a variety of inference, explanation, and tutoring tasks. Second, numerical methods optimize parameter values of distributed control problems, while SA solves for structural design problems, specifying control configurations including the number and locations of controls. Third, SA performs local computation on networks of spatial objects, without the need to construct an explicit, global model. Numerical methods typically require a global model for a field. Finally, SA encapsulates many pieces of domain-specific knowledge in its generic operators, while hiding low-level details.

A Spatial Aggregation Model of Physical Fields

Fields abstract a wide range of continuous, distributed physical phenomena such as temperature, velocity, and image data. Formally, a field maps one continuum to another. A temperature field in a room maps a location to temperature, i.e., $R^3 \to R^1$. Many important reasoning and controlling tasks for physical fields require compact structural and behavioral descriptions of the fields.

Consider a problem from manufacturing: heat treatment of a metal sheet (Jaluria & Torrance 1986), as shown in Figure 1[1]. As part of the manufacturing process, the temperature distribution over the sheet must be regulated at some desired profile over a period of time to minimize damage to the material. To describe the thermal process in the metal sheet, we need to specify physical laws, geometry, material properties, and boundary conditions. In addition, the control objective specifies a desired thermal profile over the material.

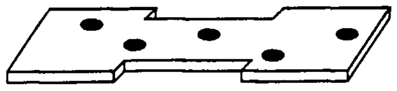

Figure 1: Industrial heat treatment of a metal sheet. The control objective is to heat the material to a desired temperature at a small number of locations, shown as dark circles, so as to minimize temperature fluctuation across the material.

Physical laws governing fields are described by partial differential equations. For instance, the physical process of isotropic, steady-state heat diffusion is governed by the Poisson equation:

$$k\nabla^2\phi + \dot{Q} = 0 \qquad (1)$$

where ∇^2 is the Laplace operator, ϕ is the temperature, k is material conduction coefficient, and $\dot{Q}$ is the source value representing heat per unit time and volume. The Poisson equation also governs other physical phenomena such as gravity and electrostatics. In fluid dynamics, the motion of fluid is governed by the Navier-Stokes equation, another well-known partial differential equation. Traditionally, detailed, labor-intensive numerical simulations are employed to elaborate consequences of these models. In contrast, our objective is to construct a qualitative physics model for spatial physical fields so that behaviors of the fields can be inferred using a small number of operations on a discrete representation and explained in terms of object interaction and evolution.

A physical field exhibits multiple temporal and spatial scales due to physical properties of the domain, geometry, boundary conditions, and external forces. Mechanical vibrations travel at the speed of sound in a piece of material. Temperature decays with varying rates along different directions, determined by material property and geometric shape. Acoustic waves mix and reflect, according to wave lengths and boundary conditions. The scales and locality permit a continuous, distributed parameter field to be abstracted as a set of discrete objects by suppressing irrelevant details. Consequently, the field can be understood by inter-

[1]A similar problem arises from temperature control in semiconductor manufacturing, in which the oven temperature over a surface of semiconductor wafer must be regulated by a set of spatially distributed heating lamps to ensure high yield. This is a challenging control problem in rapid thermal processing (RTP) of semiconductor wafers because temperature non-uniformity often leads to chip defects (Kailath & others 1996).

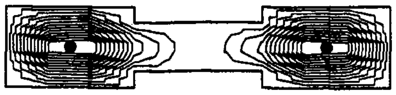

Figure 2: Temperature distribution in the dumbbell-shaped material, heated at the centers of heavy ends. The temperature gradient is described by isotherms.

preting a compact structural and behavioral representation. For instance, after examining the thermal field in dumbbell-shaped material heated at the heavy ends (Figure 2), we discover that the isotherms are more sparse along the narrow channel of the dumbbell, indicating a very small temperature decay. Intuitively, the heat flux is constrained by the narrow geometry of the channel. More formally, physics tells us this direction corresponds to the weakest coupling direction of the temperature field[2]. In order to control the field using a collection of decentralized sources, the field must be decoupled into constituent regions. We need to develop abstractions and operators that formalize the physical intuition so as to explicitly model field granularity and permit principled and programmed trade-offs among different design choices.

SA provides the **field ontology** that represents a physical field as a network of spatial objects (Yip & Zhao 1996; Bailey-Kellogg, Zhao, & Yip 1996). Each spatial object comprises a geometric description for its spatial extent and a feature description for its values. For instance, a spatial object in a temperature field describes a location and the temperature at that location. The spatial objects are governed by interaction rules relating objects to each other, along with constraints on feature values. The major elements of the SA ontology are summarized in Table 1. The field ontology is specialized for the control and optimization tasks discussed in this paper. We deploy two particularly useful field discretizations that represent a field either as a grid of points (Figure 3(a)) or as a triangular mesh of elements (Figure 3(b)). Each spatial object abstracts geometry and physics over a finite region. A neighborhood graph encodes spatial adjacencies among the objects. Local constraints, derived from a finite difference approximation to the Laplace operator at grid points or conservation properties over elements in a mesh, govern the evolution of the objects. The field values are determined by a local relaxation method that iteratively updates spatial objects using the local constraints. With these two field partitions, the SA abstraction mechanism subsumes the finite difference and finite element models commonly used in engineering and science for numerically approximating a field (Vichnevetsky 1981). We note that SA supports many important reasoning and controlling tasks

- Spatial Objects
  - Geometric description

  - Feature description
    Examples: temperature; pressure; velocity
- Constitutive Laws
  Examples:
  - Fourier's law: heat_flux $= -k\frac{dT}{dx}$
  - Ohm's law: charge_flux $= -\gamma\frac{dV}{dx}$
  - Hooke's law: stress $= E\frac{du}{dx}$
- Spatial Neighborhood Structures
  Examples:
  - Minimal spanning tree
  - Regular grid neighborhood graph
  - Delaunay mesh neighborhood graph
- Interaction Rules
  Examples:
  - Causal interaction
  - Consistency rules
  - Update rules

Table 1: Spatial Aggregation field ontology.

for physical fields, in addition to approximating the field. SA can also take advantage of existing numerical analysis software[3].

Building upon the spatial object abstraction, SA provides an aggregation-disaggregation mechanism for reasoning about structures in control and optimization of physical fields. Figure 4 illustrates the bi-directional mapping between lower- and higher-level objects within each spatial aggregation layer. SA employs a sequence of such bi-directional mappings to mediate the input fields and higher-level compact descriptions. In the aggregation process, SA operators `aggregate`, `classify`, and `redescribe` build a hierarchy of increasingly more abstract spatial object neighborhood graphs (N-graphs) and equivalence classes. During disaggregation, the operators **`disaggregate`** and **`flatten`** open up higher-level aggregate objects to permit control over finer component objects of the field. The `interpolate` operator transfers information between coarser and finer objects[4]. Additional operators act on the objects, fields, and N-graphs, to search, map, filter, update, form correspondences, and maintain consistency.

For control and optimization tasks, the input to SA

[2]To be precise, this gives the smallest positive eigenvalue of the Laplacian graph of the field (Spielman & Teng 1996).

[3]We have already incorporated the widely used numerical linear algebra software package LAPACK in SAL.

[4]The new SA operators **`disaggregate`**, **`interpolate`**, and **`update`** are introduced here for the purpose of control.

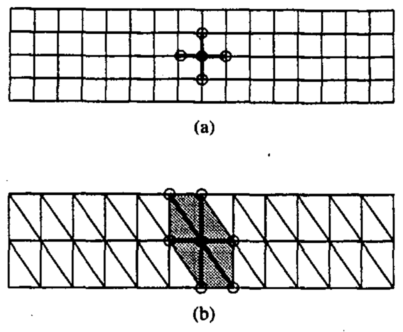

Figure 3: A field modeled by a network of local spatial objects: (a) Finite difference grid; (b) Finite element mesh. The grid or mesh defines spatial adjacency between a spatial object (solid circle) and its neighbors (empty circles). The shaded region in the mesh represents spatial extent of the object. Higher-dimensional fields can be likewise modeled.

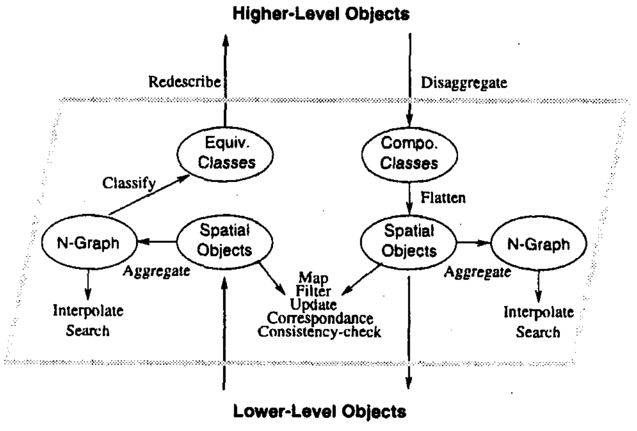

Figure 4: Spatial aggregation and disaggregation support bidirectional mapping between lower- and higher-level spatial objects. SA aggregates lower-level objects to form a neighborhood graph, classifies them into equivalence classes, and redescribes the classes as higher-level objects. Reciprocally, SA disaggregates higher-level objects into component objects and forms a neighborhood graph of the objects.

is a physical field modeled as a collection of spatial objects, a control objective, and a set of optimality constraints. SA determines a spatially distributed control that effectively steers the physical process to meet the desired criteria[5]. To determine the structure and parameters of the distributed control, i.e., the number, location, and values of control sources, one needs to search the large design space subject to performance constraints. Although many numerical methods exist for parametric design problems that optimize control values, structural design that determines the number and locations of distributed control has remained an ad hoc practice.

Illustration: Control of a Thermal Field

As an example of how spatial aggregation operators support modular programming for control applications, consider the temperature regulation problem for a rectangular piece of material, similar to that of Figure 1. The temperature field is represented by thermal spatial objects (`therm_obj`) indicating locations and temperature values. The control objective is to establish a uniform temperature distribution over the entire field, using a small set of discrete heat sources, subject to constraints on the number of control sources, maximum source output, and acceptable temperature fluctuations. This global control objective can be formulated locally by constraining each thermal object to have a temperature within some error tolerance of its desired temperature. The available control authority consists of point sources, each of which regulates temperature in a local neighborhood. The design task is to determine the number, locations, and values of heat sources required to satisfy the objective. A separate field represents control sources with spatial objects (`src_obj`) indicating locations, control values, and (possibly overlapping) regions of temperatures that each is attempting to control. Neighborhood graphs explicate spatial adjacencies among thermal objects and among source objects.

The control design is accomplished by an iterative aggregation-disaggregation process. Given a source configuration, aggregation groups sources according to locality and similarity in control authority, and replaces a group with a more global source if the replacement adequately controls the combined control regions. Disaggregation splits a single source that does not adequately control its region, replacing it with multiple more localized sources that better control subregions. An aggregation-based approach repeatedly aggregates sources, starting from a dense set, while a disaggregation-based approach repeatedly disaggregates sources, starting from a sparse set. The trade-offs

[5]A dual problem concerns placement of sensors for maximum observability. We will only consider the controllability problem in this paper and note that the observability problem can be likewise solved.

```
while control error is too large
  for each unsatisfactory control region r
    disaggregate:
      generate candidate control partitions:
        find weak coupling directions
        partition r
        optimize control locations/values
      replace control with best configuration
    update field values
```

Figure 5: Algorithm for disaggregation-based control configuration design.

between aggregation and disaggregation include the type of knowledge required (initial configurations; how to group vs. how to split) and the complexity (potential amount of work at each level; fan-in vs. fan-out). An even more advanced strategy interleaves aggregation and disaggregation so as to exploit the advantages of both.

The spatial aggregation language provides **aggregate**, **classify**, and **redescribe** operators for hierarchically building source aggregates (Bailey-Kellogg, Zhao, & Yip 1996). For brevity, we omit the code for aggregation-based design, concentrating instead on the disaggregation-based approach.

Figure 5 outlines the algorithm for disaggregation-based design. Disaggregation-based design is an iterative process, repeatedly checking the control error and disaggregating unsatisfactory sources. To disaggregate a source, multiple possible disaggregations are considered, and the one with the best error profile is chosen. To form disaggregations, we make use of the physical knowledge discussed in the preceding section and illustrated in Figure 2: small gradients indicate weak coupling. By placing new sources along directions of weak coupling, a design can optimally "divide-and-conquer" the problem. The remainder of this section illustrates how the spatial aggregation language supports the implementation of this algorithm, providing operators at the right level of abstraction and encapsulating the domain-specific physical knowledge.

The spatial aggregation operator update specifies iterative processes by repeatedly applying user-specified update rules until convergence is reached. The update rule for source disaggregation checks whether a given source adequately controls its region and disaggregates it if the error is too large (see Figure 6(a)). Note that the algorithm presented here is greedy, but could be extended to maintain multiple possibilities in a search tree, or to allow backtracking through a dependency network.

The disaggregate operator replaces a spatial object with a set of constituent objects, exploring the structure of the domain and taking advantage of locality

```
update_source_by_disaggregation(src_obj)
  if error(region(src_obj)) > max_error
    new_src_objs = disaggregate(src_obj)
    replace(source_field, src_obj,
            new_src_objs)
```
(a)

```
partition_ctrl_region(src_obj)
  ctrl_reg = region(src_obj)
  if area(ctrl_reg) < min_area
    return empty_set
  else
    subregs = decouple_along_gradient(ctrl_reg)
    new_src_obj_sets =
      map(subregs,
          create_src_objs_at_sample_points)
    return
      create_sets_with_one_member_from_each
        (new_src_obj_sets)
```
(b)

```
update_source_by_gradient_descent(src_obj)
  grad = gradient(error_in_region, src_obj)
  if abs(grad) > min_change
    new_ctrl_val =
      ctrl_value(src_obj) + step_size*grad
    set_ctrl_value(src_obj, new_ctrl_val)
```
(c)

```
update_value_by_fd(therm_obj)
  old_temp = temp(therm_obj)
  new_temp =
    avg(neighbors(therm_ngraph, therm_obj))
    + alpha*source_at(position(therm_obj)))
  if abs(old_temp - new_temp) > min_change
    set_temp(therm_obj, new_temp)
```
(d)

```
refine(therm_obj)
  for_each W, S, SW nbr in
      neighbors(therm_ngraph, therm_obj)
    new_pos = midpoint(position(therm_obj),
                       position(nbr))
    new_temp = interpolate(new_pos, therm_ngraph,
                           temperature))
    add(therm_field,
        create_therm_obj(new_pos, new_temp))
```
(e)

Figure 6: User-level SA pseudocode for disaggregation-based temperature regulation design. (a) Update rule for disaggregating an unsatisfactory source. (b) Generation of partitions for disaggregation, using physical knowledge. (c) Update rule for optimizing source value. (d) Update rule for evaluating thermal field using finite difference approximation. (e) Update rule for multigrid-style refinement of temperature field.

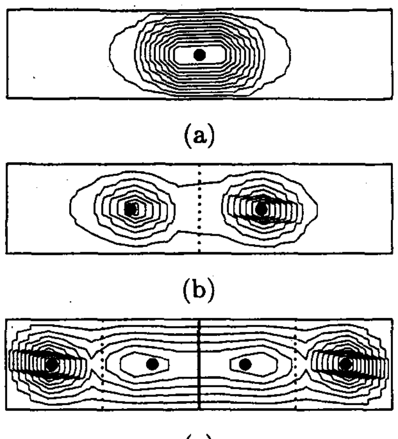

Figure 7: Control a thermal field by successive source disaggregations, exploiting gradient information. Dark circles indicate source positions, vertical lines delineate regions of control, and curves describe isotherms.

and scales determined by geometry and physics. Different implementations use different user-specified knowledge to help disaggregate an object. In Figure 6(b), a user-defined partition function divides a source's control region into subregions, decoupling along the gradient as discussed above. It then places new sources at various points in the new regions and forms the cross product of the sets of possible new sources, returning sets of possible disaggregations with one new source for each subregion. `Disaggregate` tests the resulting error in each partition to choose an optimal disaggregation for a source. Figure 7 shows source locations and isotherms after the first few disaggregations for the temperature regulation problem. The decoupling algorithm used here splits in half along the weaker coupling axis when a source does not adequately control its region. More advanced disaggregation would partition the field using more refined knowledge of gradients and other scales; the next section illustrates how spatial aggregation operators can support such knowledge.

The initial values of the new sources in Figure 6(b) can be determined by proportioning the parent source's value according to the areas. An alternative is to approximate the required source density by applying finite differences to the control region, and then scaling by the area the source covers. In either case, the initial guess can likely be improved. A local adjustment method, such as gradient descent, can be specified in terms of an update rule. Figure 6(c) provides code for an update rule to adjust the value of a source in order to minimize the error in its control region. The position of a source can be likewise adjusted.

To measure the quality of control, the temperature field is computed for the specified source configuration. A local relaxation method is specified by the `update`

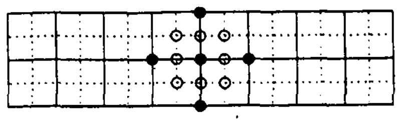

Figure 8: Spatial object refinement using a finite difference grid approximation. New objects (empty circles) are inserted between old objects (solid circles); dotted lines show the finer grid.

operator. The rule in Figure 6(d) updates a thermal object based on values of neighbors and any source located at that point. Different implementations of `update` update objects sequentially or in parallel; in this case, that yields the Gauss-Seidel or Jacobi numerical relaxation method.

The computational efficiency of the temperature solver could be improved by a multi-level method (Briggs 1987) that uses a solution for coarse spatial objects as an initial guess for a solution for finer spatial objects, as shown in Figure 8. The code in Figure 6(e) refines a coarse thermal object, adding points between the object and its west, south, and southwest neighbors in a finite difference grid. The `interpolate` operator fills in temperatures for the new thermal objects based on values of neighboring objects in the grid. Various implementations of `interpolate` support linear interpolation, quadratic interpolation, and so forth.

In summary, the spatial aggregation operators provide powerful building blocks for developing programs for distributed control problems of thermal fields. The operators modularize user-specified knowledge and abstract away many low-level implementation details. Table 2 summarizes the SA operators and their implementations currently available in the C++-based compiler for SAL. The disaggregation-based design program runs reasonably efficiently in SAL: using a 500-object field discretization, 8 candidate source locations per disaggregation, and fairly strict convergence criteria, it takes 1 to 10 minutes on a 100MHz Pentium to design the source configuration plotted in Figure 7, depending on the constraints placed on source values.

A More Complex Example

As we have illustrated for a simple rectangular domain, the spatial aggregation-disaggregation approach to decentralized control exploits the locality of a field to find a rough solution by divide-and-conquer and then locally refine the control placement and actions. Controlling complex spatial domains requires more detailed knowledge about the field. In particular, SA operators are employed to extract information about the thermal gradient.

For example, consider the problem in Figure 9(a): an irregularly-shaped metal sheet has a hole near its right end and a single heat source near the center of mass. Figure 9(b) shows temperature gradient field di-

- **Disaggregate**: object → objects
  Refines `object` into a set of component `objects`.
  - Generate a single refinement, using user-provided generator
  - Enumerate multiple refinements and choose the best one, using user-provided generator and selector
  - Generate multiple refinements as needed and choose the first satisfactory one, using user-provided stream generator and tester
- **Interpolate**: geometry * objects * value_function → value
  Determines `value` for `geometry` based on values of `objects` given by `value_function`.
  - Average of members of a set
  - Locally weighted average in a `field`
  - Average of neighbors in an `ngraph`
- **Update**: objects * update_rule → objects
  Modifies `objects` by repeatedly applying `update_rule` to each.
  - Object collection: a set or field
  - Temporal order: sequential or parallel
  - Convergence criteria: fixed number of times or until predicate satisfied
- Existing SA data types and operators:
  - **Spatial object, field, ngraph**
  - **Aggregate, classify, redescribe, search, map, filter**

Table 2: SA data types and operators.

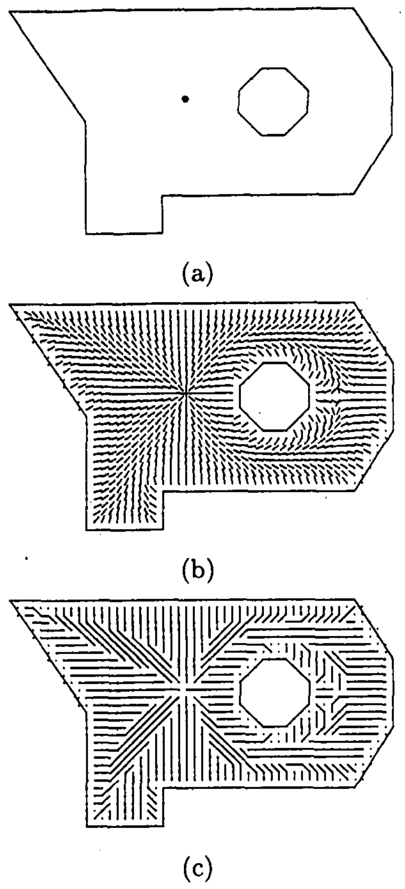

Figure 9: A complex thermal domain: (a) Geometry and heat source location; (b) Directions of the resulting temperature gradient field; (c) Gradient trajectories aggregated by SA operators.

rections (a gradient vector is normal to the isotherm curve) computed from a pointwise description of the temperature. The gradient vectors are then grouped into equivalence classes and redescribed as trajectories using the SA operators `classify` and `redescribe`, as shown in Figure 9(c). Average gradient magnitudes along trajectories through the source are compared to find directions of weaker decay; the `disaggregate` operator splits source objects along such directions as previously discussed.

Discussion

The SA aggregation-disaggregation mechanism modularizes common computational ideas in data interpretation and control tasks and permits reuse of generic operators. Using these operators, a problem can be described at a level closer to one's intuitive understanding of the physics. The resulting programs are easier to understand and modify. The multi-layer spatial aggregates can be used for automatically generating explanations for why higher-level control decisions are made. For instance, the field geometry might induce a weak gradient direction; hence, SA refines spatial objects along that direction in order to decouple the field. In contrast, traditional numerical simulations require humans to interpret and explain the results.

Traditionally, qualitative physics has focused on lumped parameter models of physical systems (DeKleer & Brown 1984; Forbus 1984; Kuipers 1986). For instance, the device ontology describes an electrical circuit as a network of components, while abstracting away the spatial dimension. However, many important physical phenomena are modeled as spatially distributed parameter systems. The SA field ontology describes such physical phenomena by encapsulating the important spatial information in the spatial objects and the neighborhood structures, permitting efficient reasoning about these phenomena.

Unlike control design for lumped parameter linear systems, few analytic design techniques have been developed for distributed control of large physical fields (Sandell Jr. *et al.* 1978). In practice, the design is often accomplished by brute-force numerical simulations. SA offers a powerful modeling framework and an alternative mechanism for synthesizing decentralized control. The multi-resolutional SA model is particularly useful for formulating structural design problems that are not well addressed by numerical methods. In addition, the SA mechanism can integrate existing numerical techniques to solve parametric design problems.

We believe SA is applicable to a wide variety of problems in data interpretation and control. For instance, the parti-game algorithm for learning control strategies in high-dimensional state-spaces can be formulated in terms of SA operators (Moore & Atkeson 1995). Similarly, Bradley and Zhao presented several methods for synthesizing nonlinear control laws in phase spaces (Bradley & Zhao 1993); their methods partition phase spaces into manageable subspaces. SA can also describe adaptive aggregation in dynamic programming where states and their dependencies are grouped into meta-states according to residual errors (Bertsekas & Castanon 1989).

We have described steady-state control problems for a class of physical fields such as heat conduction, gravity, electrostatics, and incompressible fluid flow that are governed by elliptic partial differential equations. To apply SA to transient problems, we need to develop techniques for tracking and correlating temporal objects. Another important class of problems involves wave phenomena (such as sound) governed by hyperbolic partial differential equations. Extending SA to address wave control problems remains as a future research topic.

Other researchers have developed related frameworks and systems for reasoning about spatial, analogue representations of the physical world. Williams and Nayak discussed model-based methods for modeling, configuring and programming distributed physical systems (Williams & Nayak 1996). Lundell presented a qualitative model for distributed parameter physical fields (Lundell 1996). Forbus et al. developed the Metric Diagram/Place Vocabulary (MD/PV) framework for qualitative spatial reasoning (Forbus, Nielsen, & Faltings 1991). In computer scene analysis, Hummel and Zucker formalized a class of problems called relaxation labeling that computes a consistent labeling for a network of objects (Hummel & Zucker 1983). In comparison to the above work, the SA framework focuses on structure recovery and control of spatial fields. It differs from relaxation labeling in that SA builds multi-layer spatial aggregates to exploit a variety of spatial and temporal constraints. It also differs from many neural net based learning and optimization methods in that SA explicitly models and exploits field topological structures. This paper develops SA mechanisms for controlling physical fields that significantly extend the framework developed in (Yip & Zhao 1996; Bailey-Kellogg, Zhao, & Yip 1996).

Conclusions

This paper advances the state-of-the-art in qualitative physics and spatial reasoning in several ways. It has developed the SA ontological abstraction for distributed parameter physical fields and an aggregation-disaggregation mechanism for synthesizing structures and parameters for distributed control of the fields. It has illustrated how a modular program can be written using the SA generic operators for the control and optimization of a thermal field regulation problem. SA extracts meaningful structures from continuous fields so that design decisions can be articulated using discrete spatial objects. Furthermore, because spatial aggregates comprise mixed numeric, geometric, and symbolic descriptions, we expect SA to take advantage of and complement the arsenal of existing numerical methods.

This research addresses several scientific and engineering concerns central to artificial intelligence. We expect SA to shed light on bi-directional mappings between macroscopic decisions and local actions in biological and engineered systems. In addition, SA aims to systematize design principles and programming methodologies for interpreting and controlling distributed parameter physical fields, and to provide useful tools for practicing engineers.

Acknowledgment

We thank the following people for helpful discussions: Andy Berlin, Geoff Chase, Tad Hogg, Xingang Huang, Shiou Loh, Jeff May, Pandu Nayak, Ivan Ordonez, Elisha Sacks, Gerry Sussman, Brian Williams, and Ken Yip.

This research is supported in part by an NSF National Young Investigator Award CCR-9457802, an Alfred P. Sloan Foundation Research Fellowship, a grant from Xerox Palo Alto Research Center, and an AT&T grant.

References

Bailey-Kellogg, C.; Zhao, F.; and Yip, K. 1996. Spatial aggregation: language and applications. In *Proceedings of AAAI.*

Berlin, A. 1994. MEMS-based active structural strengthening technology. In *Proceedings of Government Microcircuit Apllications Conference.*

Bertsekas, D., and Castanon, D. 1989. Adaptive aggregation methods for infinite horizon dynamic programming. *IEEE Trans. on Automatic Control* 34.

Bradley, E., and Zhao, F. 1993. Phase-space control system design. *IEEE Control Systems* 13(2):39-47.

Briggs, W. 1987. *A Multigrid Tutorial.* Lancaster Press.

DeKleer, J., and Brown, J. 1984. A qualitative physics based on confluences. *Artificial Intelligence* 24.

Forbus, K.; Nielsen, P.; and Faltings, B. 1991. Qualitative spatial reasoning: the CLOCK project. *Artificial Intelligence* 51.

Forbus, K. 1984. Qualitative process theory. *Artificial Intelligence* 24.

Hummel, R., and Zucker, S. 1983. On the foundations of relaxation labeling processes. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 5(3).

Jaluria, Y., and Torrance, K. 1986. *Computational Heat Transfer.* Hemisphere Publishing.

Kailath, T., et al. 1996. Control for advanced semiconductor device manufacturing: A case history. In Levine, W., ed., *The Control Handbook.* CRC Press.

Kuipers, B. 1986. Qualitative simulation. *Artificial Intelligence* 29.

Lundell, M. 1996. A qualitative model of physical fields. In *Proceedings of AAAI.*

Moore, A., and Atkeson, C. 1995. The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning* 21.

Sandell Jr., N.; Varaiya, P.; Athans, M.; and Safonov, M. 1978. Survey of decentralized control methods for large scale systems. *IEEE Trans. on Automatica Control* 23(2).

Seidman, T. 1996. Control of the heat equation. In Levine, W., ed., *The Control Handbook.* CRC Press.

Spielman, D., and Teng, S. 1996. Spectral partitioning works: Planar graphs and finite element meshes. Technical Report UCB CSD-96-898, UC Berkeley.

Vichnevetsky, R. 1981. *Computer Methods for Partial Differential Equations, Vol. 1.* Prentice-Hall.

Williams, B., and Nayak, P. 1996. Immobile robots: AI in the new millenium. *AI Magazine* 17(3).

Yip, K. M., and Zhao, F. 1996. Spatial aggregation: theory and applications. *J. Artificial Intelligence Research* 5.