Advanced Vision (INF11031)

Current learning outcomes:

1. understand machine vision principles (assessed by exam).
2. be able to acquire and process raw image data (assessed practical).
3. be able to relate image data to 3D scene structures (assessed practical).
4. know the concepts behind and how to use several model-based object representations, and to critically compare them (assessed by exam).
5. know many of the most popularly used current computer vision techniques (assessed by exam).
6. undertake computer vision work in MATLAB (assessed practical).

Proposed learning outcomes:

1. understand machine vision principles (assessed by exam).
2. be able to acquire and process raw image data (assessed practical) and to relate image data to 3D scene structures (assessed practical).
3. know the concepts behind and how to use several model-based object representations, and to critically compare them (assessed by exam).
4. know many of the most popularly used current computer vision techniques (assessed by exam).
5. undertake computer vision work in MATLAB (assessed practical).

Agent Based Systems (INF10049)

Course Change due to UG3 restructuring (10 and 20-credit point versions)

Automated Reasoning (INF11074)

Course Change due to UG3 restructuring. Done.

Computational Complexity (INF11102)

Current Learning Outcomes:

1. Students will be able to formulate computational models with resource constraints, and be able to describe relationships between these models.
2. Students will be able to analyse computational problems from a complexity perspective, and so locate them within the complexity landscape.
3. Students will be able to apply mathematical skills and knowledge from earlier years (eg., from logic and discrete mathematics) to concrete problems in computational complexity.
4. Students will gain an appreciation of the broader importance of fundamental problems in computer science, such as the P vs NP problem.

No change.

Computer Algebra (INF10009)

Current Learning Outcomes:

1. Use the computer algebra system Maple as an aid to solving mathematical problems.
2. Design and implement in Maple appropriate algorithms from constructive mathematical solutions to problems.
3. Discuss the overall design of the computer algebra system Maple.
4. Evaluate the results obtained from a computer algebra system and discuss possible problems.
5. Explain the gap between ideal solutions and actual systems (the need to compromise for efficiency reasons).
6. Describe and evaluate data structures used in the computer representation of mathematical objects.
7. Discuss the mathematical techinques used in the course and relate them to computational concerns.
8. Discuss and apply various advanced algorithms and the mathematical techniques used in their design.
9. Use the techniques of the course to design an efficient algorithm for a given mathematical problem (of a fairly similar nature to those discussed in the course).

Proposed Learning Outcomes:

1. Use the computer algebra system Maple as an aid to solving mathematical problems, and design and implement in Maple appropriate algorithms from constructive mathematical solutions to problems
2. Discuss the overall design of the computer algebra system Maple and evaluate the results obtained from a computer algebra system and discuss possible problems.
3. Explain the gap between ideal solutions and actual systems (the need to compromise for efficiency reasons).
4. Describe and evaluate data structures used in the computer representation of mathematical objects, and discuss the mathematical techniques used in the course and relate them to computational concerns.
5. Discuss and apply various advanced algorithms and the mathematical techniques used in their design, and use the techniques of the course to design an efficient algorithm for a given mathematical problem (of a fairly similar nature to those discussed in the course).

Computer Architecture (INF09009)

Current Learning Outcomes:

1. Demonstrate the ability to describe the structure and operational characteristics of a pipelined microprocessor.
2. Demonstrate the ability to explain principles of: orthogonal instruction set design; pipeline hazards and interlocks; out of order execution; scoreboards and reservation stations and their use; branch prediction (both static and dynamic); and techniques (both software and hardware) for exploiting loop-level parallelism.
3. Demonstrate the ability to quantitatively evaluate the performance of a combined processor and memory system with respect to cycles-per-instruction (CPI) and memory bandwidth requirements.
4. Demonstrate the ability to explain the principle of memory locality, to show how a memory hierarchy exploits the various forms of locality, and to analyze the performance of a memory hierarchy.
5. Demonstrate the ability to design, in outline, a memory hierarchy, and to specify reasonable parameters for each configuration point (capacity, associativity, block size, and write policies) at each level in the hierarchy.
6. Demonstrate an understanding of the memory coherency issues involved when designing a multiprocessor system, and to explain the behaviour of a typical cache coherency protocol.

Proposed Learning Outcomes:

1. Demonstrate the ability to describe the structure and operational characteristics of a pipelined microprocessor, and to explain principles of: orthogonal instruction set design; pipeline hazards and interlocks; out of order execution; scoreboards and reservation stations and their use; branch prediction (both static and dynamic); and techniques (both software and hardware) for exploiting loop-level parallelism.
2. Demonstrate the ability to quantitatively evaluate the performance of a combined processor and memory system with respect to cycles-per-instruction (CPI) and memory bandwidth requirements.
3. Demonstrate the ability to explain the principle of memory locality, to show how a memory hierarchy exploits the various forms of locality, and to analyze the performance of a memory hierarchy.
4. Demonstrate the ability to design, in outline, a memory hierarchy, and to specify reasonable parameters for each configuration point (capacity, associativity, block size, and write policies) at each level in the hierarchy.
5. Demonstrate an understanding of the memory coherency issues involved when designing a multiprocessor system, and to explain the behaviour of a typical cache coherency protocol.

Computer Graphics (INF11021)

Current Learning Outcomes:

1. Analyse and synthesise algorithms for the display of antialiased lines in Euclidean space
2. Analyse and synthesise algorithms for the display of space curves and surfaces of arbitrary smoothness
3. Model any arbitrary shape of 3D object and to perform combinations of affine transformations on these objects in 3D space
4. Construct views from arbitrary viewpoints in space and project to an image plane
5. Order objects from the viewpoint in order to perform visible surface computations
6. Use surface properties of objects and scene illuminations to produce realistic images modelling the interaction of light and objects within a scene
7. Calculate the shadows based on the relationships of the lights and objects
8. Simulate the movement of photons to compose realistic images

Proposed Learning Outcomes:

1. Analyse and synthesise algorithms for the display of anti-aliased lines in Euclidean space, and of space curves and surfaces of arbitrary smoothness
2. Model any arbitrary shape of 3D object and to perform combinations of affine transformations on these objects in 3D space
3. Construct views from arbitrary viewpoints in space and project to an image plane, and order objects from the viewpoint in order to perform visible surface computations
4. Use surface properties of objects and scene illuminations to produce realistic images modelling the interaction of light and objects within a scene
5. Calculate the shadows based on the relationships of the lights and objects, and simulate the movement of photons to compose realistic images

Computer Science Large Practical (INF09040)

Course Change due to UG3 restructuring

Database Systems (INF10055)

Course Change due to UG3 restructuring

Distributed Systems (INF11022)

Current Learning Outcomes:

1. have an understanding of the principles of distributed system and be able to demonstrate this by explaining them;
2. be able to give an account of the trade-offs which must be made when designing a distributed system, and make such trade-offs in their own designs;
3. have developed practical skills of implementation of distributed algorithms in software so that they will be able to take an algorithm description and realise it in software;
4. be able to give an account of the theoretical models used to design distributed systems and to manipulate those models to reason about such systems.
5. be able to design efficient algorithms for distributed computing tasks.
6. This course will teach students the principles of distributed systems, in particular those aspects which make such systems difficult to design and develop.

Proposed Learning Outcomes:

1. Develop an understanding of the principles of distributed system and be able to demonstrate this by explaining them;
2. Being able to give an account of the trade-offs which must be made when designing a distributed system, and make such trade-offs in their own designs;
3. Develop practical skills of implementation of distributed algorithms in software so that they will be able to take an algorithm description and realise it in software;
4. Being able to give an account of the theoretical models used to design distributed systems and to manipulate those models to reason about such systems.
5. Being able to design efficient algorithms for distributed computing tasks.

Human Computer Interaction (INF11017)

Current Learning Outcomes:

1. Demonstrate, in writing, knowledge of the issues and problems in HCI.
2. Demonstrate an understanding of human perception and behaviour in analysing their interactions with technology in their every day lives.
3. Use established design principles and methodologies to solve HCI problems.
4. Acquire confidence in handling different disciplinary perspectives on HCI and the ability to apply them to design problems.
5. The ability to devise, plan and execute task analysis and system evaluation studies from an HCI perspective, and present findings in a clear and effective manner.
6. Demonstrate awareness of current areas of research by locating and summarising examples of recent progress.

Proposed Learning Outcomes:

1. Demonstrate, in writing, knowledge of the issues and problems in HCI, and an understanding of human perception and behaviour in analysing their interactions with technology in their every day lives.
2. Use established design principles and methodologies to solve HCI problems.
3. Acquire confidence in handling different disciplinary perspectives on HCI and the ability to apply them to design problems.
4. The ability to devise, plan and execute task analysis and system evaluation studies from an HCI perspective, and present findings in a clear and effective manner.
5. Demonstrate awareness of current areas of research by locating and summarising examples of recent progress.

Informatics 1 – Data and Analysis (INF08015)

Current Learning Outcomes:

1. Demonstrate knowledge of the terminology and paradigms used in different areas of informatics for collecting, representing and interpreting data, by being able to apply them to sample problems.
2. Demonstrate understanding of different types of data (for example, structured/semistructured/unstructured, quantitative/qualitative).
3. Demonstrate proficiency of the entity/relationship model by being able to specify appropriate representations and queries for simple examples.
4. Show awareness of the importance of logic for the representation of data by being able to design simple logical representation of a given data set.
5. Present data in a variety of forms (textual, graphical, quantitative), across a range of data types.
6. Show awareness of the distinction between object data and meta-data, by being able to apply it to a number of applications across informatics (e.g., databases, corpora).
7. Demonstrate knowledge of the basic algorithms for interpreting and processing data, by being able to demonstrate how these algorithms work for simple data sets.

Proposed Learning Outcomes:

1. Demonstrate knowledge of the terminology and paradigms used in different areas of informatics for collecting, representing and interpreting data, by being able to apply them to sample problems.
2. Demonstrate understanding of different types of data (for example, structured/semistructured/unstructured, quantitative/qualitative), and of the proficiency of the entity/relationship model by being able to specify appropriate representations and queries for simple examples.
3. Show awareness of the importance of logic for the representation of data by being able to design simple logical representation of a given data set, and to present data in a variety of forms (textual, graphical, quantitative), across a range of data types.
4. Show awareness of the distinction between object data and meta-data, by being able to apply it to a number of applications across informatics (e.g., databases, corpora).
5. Demonstrate knowledge of the basic algorithms for interpreting and processing data, by being able to demonstrate how these algorithms work for simple data sets.

Informatics 1 – Object-oriented Programming (INF08014)

Current Learning Outcomes:

1. Name, explain and apply the core concepts and constructs used in imperative and object-oriented programming.
2. Given a detailed design, develop a working program that implements the design.
3. Develop small programs, or components of larger ones, or modify existing ones, to solve clearly defined programming problems.
4. Given a clearly described component, develop a test set and test code for a component. Use code review and debugging tools to identify the location of a fault in an erroneous program.
5. Run and analyse a given program; describe how well it works compared to its specification, or identify ways in which it fails.
6. Apply basic tools to aid in developing programs (e.g. IDE, version control).

Proposed Learning Outcomes:

1. Name, explain and apply the core concepts and constructs used in imperative and object-oriented programming.
2. Given a detailed design, develop a working program that implements the design.
3. Develop small programs, or components of larger ones, or modify existing ones, to solve clearly defined programming problems.
4. Given a clearly described component, develop a test set and test code for a component. Use code review and debugging tools to identify the location of a fault in an erroneous program.
5. Run and analyse a given program; describe how well it works compared to its specification, or identify ways in which it fails, and apply basic tools to aid in developing programs (e.g. IDE, version control).

Informatics 2B – Algorithms, Data Structures, Learning (INF08009)

Current Learning Outcomes:

1. Demonstrate the ability to analyse the complexity of algorithms using asymptotic notation.
2. Demonstrate the ability to write programs to create and manipulate array-structured and dynamically-structured data.
3. Demonstrate the ability to construct and analyse search tree data structures.
4. Demonstrate knowledge of sorting algorithms and their run-time complexity
5. Demonstrate knowledge of graph algorithms
6. Demonstrate understanding of statistical pattern recognition and Bayes theorem
7. Demonstrate the ability to manipulate and describe multidimensional data using summary statistics.
8. Demonstrate the ability to model discrete multidimensional data using Naive Bayes
9. Demonstrate the ability to classify multidimensional data using Gaussians and single-layer networks
10. Demonstrate understanding of the concept of discriminant functions
11. Demonstrate the ability to model data using nearest-neighbour and clustering approaches

Proposed Learning Outcomes:

1. Demonstrate the ability to analyse the complexity of algorithms using asymptotic notation, to write programs to create and manipulate array-structured and dynamically-structured data, and to construct and analyse search tree data structures.
2. Demonstrate knowledge of sorting and graph algorithms and their run-time complexity
3. Demonstrate understanding of statistical pattern recognition and Bayes theorem, and the ability to manipulate and describe multidimensional data using summary statistics.
4. Demonstrate the ability to model discrete multidimensional data using Naive Bayes, and the ability to classify multidimensional data using Gaussians and single-layer networks
5. Demonstrate understanding of the concept of discriminant functions, and the ability to model data using nearest-neighbour and clustering approaches

Informatics 2C – Introduction to Software Engineering (INF08019)

Current Learning Outcomes:

1. Explain how to apply commonly agreed ethical principles to a software engineering situation.
2. Motivate and describe the activities in the software engineering process.
3. Construct use cases for the system requirements.
4. Explain and construct UML class diagrams and sequence diagrams.
5. Understand and construct a software system using Java.
6. Assess the software system using testing and other appropriate tools.
7. Evaluate aspects of human usability of an application program or web site.
8. Compare different approaches to software licensing.

Proposed Learning Outcomes:

1. Explain how to apply commonly agreed ethical principles to a software engineering situation, and motivate and describe the activities in the software engineering process.
2. Construct use cases for the system requirements, and explain and construct UML class diagrams and sequence diagrams.
3. Understand and construct a software system using Java, and assess the software system using testing and other appropriate tools.
4. Evaluate aspects of human usability of an application program or web site
5. Compare different approaches to software licensing.

Informatics Research Review (INF11034)

Current Learning Outcomes:

1. Select literature appropriate for the review subject.
2. Critically evaluate research literature in the chosen area.
3. Search and use appropriately databases of scientific literature.
4. Evaluate and search traditional library resources.
5. Discuss a research topic in detail leading to new hypotheses.
6. Deliver a detailed and balanced report on a research topic.

Proposed Learning Outcomes:

1. Select literature appropriate for the review subject, and critically evaluate research literature in the chosen area.
2. Search and use appropriately databases of scientific literature.
3. Evaluate and search traditional library resources.
4. Discuss a research topic in detail leading to new hypotheses.
5. Deliver a detailed and balanced report on a research topic.

Introduction to Theoretical Computer Science (INF10059)

Current Learning Outcomes:

1. Explain decidability, undecidability and the halting problem.
2. Demonstrate the use of reductions for undecidability proofs.
3. Explain the notions of P, NP, NP-complete.
4. Use reductions to show problems to be NP-hard.
5. Write short programs in lambda-calculus.
6. Explain and demonstrate type-inference for simple programs.

Proposed Learning Outcomes:

1. Explain decidability, undecidability and the halting problem.
2. Demonstrate the use of reductions for undecidability proofs.
3. Explain the notions of P, NP, NP-complete, and use reductions to show problems to be NP-hard.
4. Write short programs in lambda-calculus.
5. Explain and demonstrate type-inference for simple programs.

MInf Project (Part 2) (INFR11093)

Current Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in Informatics.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic.
4. Discuss and solve conceptual problems which arise during the investigation.
5. Justify design decisions made during the investigation.
6. Critically evaluate the investigation.
7. Present work orally and visually, with demonstration of working artifacts where appropriate.

Proposed Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in Informatics.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic, and discuss and solve conceptual problems which arise during the investigation.
4. Justify design decisions made during the investigation, and critically evaluate the investigation.
5. Present work orally and visually, with demonstration of working artifacts where appropriate.

MSc by Research Thesis (Data Science) (INFR11106)

Current Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in the area of Data Science.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic.
4. Discuss and solve conceptual problems which arise during the investigation.
5. Justify design decisions made during the investigation.
6. Critically evaluate the investigation.
7. Present their work, with demonstration of working artifacts where appropriate.

Proposed Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in the area of Data Science.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic, and discuss and solve conceptual problems which arise during the investigation.
4. Justify design decisions made during the investigation, and critically evaluate the investigation.
5. Present their work, with demonstration of working artifacts where appropriate.

MSc by Research Thesis (Pervasive Parallelism) (INFR11109)

Current Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in the area of Pervasive Parallelism.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic.
4. Discuss and solve conceptual problems which arise during the investigation.
5. Justify design decisions made during the investigation.
6. Critically evaluate the investigation.
7. Present their work, with demonstration of working artifacts where appropriate.

Proposed Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in the area of Pervasive Parallelism.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic, and discuss and solve conceptual problems which arise during the investigation.
4. Justify design decisions made during the investigation, and critically evaluate the investigation.
5. Present their work, with demonstration of working artifacts where appropriate.

Masters Dissertation (Design Informatics) (INFR11097)

Current Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in Design Informatics.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic.
4. Discuss and solve conceptual problems which arise during the investigation.
5. Justify design decisions made during the investigation.
6. Critically evaluate the investigation.
7. Present their work, with demonstration of working products or services where appropriate.

Proposed Learning Outcomes:

1. Structure and summarise a body of knowledge relating to a substantial project topic in Design Informatics.
2. Critically evaluate previous work in the area.
3. Conduct a programme of work in further investigation of issues related to the topic, and discuss and solve conceptual problems which arise during the investigation.
4. Justify design decisions made during the investigation, and critically evaluate the investigation.
5. Present their work, with demonstration of working products or services where appropriate.

Performance Modelling (Level 11) (INFR11082)

Current Learning Outcomes:

1. Students will understand the key ideas of performance modelling and the trade-offs between timeliness and efficient use of resources. They will be able to demonstrate this by an ability to give an account of these ideas and explain why the trade-off occurs.
2. Students will know the operational laws and be able to apply them to any system which satisfies the appropriate conditions to derive further information about the system. Furthermore they will be able to assess from a system description whether the conditions are met.
3. They will have the ability to design, construct and solve a simple performance model based on a Markov process in various high-level modelling formalisms as well as directly at the state transition level. Moreover they will be able to give an account of the underlying mathematics and concepts of steady state and transient analysis. The students should understand, and be able to give an account of, the assumptions which must be made about a system in order to model it as a Markov process.
4. Students will develop a basic understanding of simulation and the difference between algorithmic and analytic modelling. They will appreciate the components of the simulation engine and the importance that they are implemented efficiently.
5. Students will develop judgement with respect to choosing an appropriate modelling technique for a given scenario, so that when given a description of a problem, and the resources and skills available, they are able to recommend the best-suited modelling formalism and solution technique.
6. Students will learn to abstract from extraneous detail and focus on the important aspects of a problem. They will also understand the importance of matching the model to the question to be answered.
7. Students will develop the ability to assimilate knowledge about different formalisms and tools and put them to practical use.
8. Students will develop skills in analysing and interpreting presented data.

Proposed Learning Outcomes:

1. Students will understand the key ideas of performance modelling and the trade-offs between timeliness and efficient use of resources. They will be able to demonstrate this by an ability to give an account of these ideas and explain why the trade-off occurs. Students will know the operational laws and be able to apply them to any system which satisfies the appropriate conditions to derive further information about the system. Furthermore they will be able to assess from a system description whether the conditions are met.
2. They will have the ability to design, construct and solve a simple performance model based on a Markov process in various high-level modelling formalisms as well as directly at the state transition level. Moreover they will be able to give an account of the underlying mathematics and concepts of steady state and transient analysis. The students should understand, and be able to give an account of, the assumptions which must be made about a system in order to model it as a Markov process.
3. Students will develop a basic understanding of simulation and the difference between algorithmic and analytic modelling. They will appreciate the components of the simulation engine and the importance that they are implemented efficiently.
4. Students will develop judgement with respect to choosing an appropriate modelling technique for a given scenario, so that when given a description of a problem, and the resources and skills available, they are able to recommend the best-suited modelling formalism and solution technique. Students will also learn to abstract from extraneous detail and focus on the important aspects of a problem, and understand the importance of matching the model to the question to be answered.

5. Students will develop the ability to assimilate knowledge about different formalisms and tools and put them to practical use. They will also develop skills in analysing and interpreting presented data.

Probabilistic Modelling and Reasoning (INFR11050)

Current Learning Outcomes:

1. Define the joint distribution implied by directed and undirected probabilistic graphical models.
2. Carry out inference ingraphical models from first principles by hand, and by using the junction tree algorithm.
3. Demonstrate understanding of maximum likelihood and Bayesian methods for parameter estimation by hand derivation of estimation equations for specific problems.
4. Critically discuss differences between various latent variable models for data.
5. Derive EM updates for various latent variable models (e.g. mixture models).
6. Define entropy, joint entropy, conditional entropy, mutual information, expected code length.
7. Demonstrate ability to design, assess and evaluate belief network models.
8. Use matlab code implementing probabilistic graphic models.
9. Demonstrate ability to conduct experimental investigations and draw conclusions from them.

Proposed Learning Outcomes:

1. Define the joint distribution implied by directed and undirected probabilistic graphical models, and carry out inference in graphical models from first principles by hand, and by using the junction tree algorithm.
2. Demonstrate understanding of maximum likelihood and Bayesian methods for parameter estimation by hand derivation of estimation equations for specific problems.
3. Critically discuss differences between various latent variable models for data, and derive EM updates for various latent variable models (e.g. mixture models).
4. Define entropy, joint entropy, conditional entropy, mutual information, expected code length.
5. Demonstrate ability to design, assess and evaluate belief network models, and use Matlab code implementing probabilistic graphic models, and demonstrate ability to conduct experimental investigations and draw conclusions from them.

Reinforcement Learning (INFR11010)

Current Learning Outcomes:

1. Knowledge of basic and advanced reinforcement learning techniques.
2. Insight into the problems involved in applying these techniques to deal with real world data, and how to overcome those problems.
3. Appreciation and identification of suitable learning tasks to which these learning techniques can be applied
4. Ability to evaluate how effective a particular learning procedure has been -- internal indicators of learning success vs. actual behaviour of the learner.
5. Use and writing of Matlab programs, ability to set up and run computational experiments to produce statistically sound results
6. Formulation of problems, evaluation of results from the student's own experiments and those presented in some cases in the research literature.

Proposed Learning Outcomes:

1. Knowledge of basic and advanced reinforcement learning techniques.
2. Insight into the problems involved in applying these techniques to deal with real world data, and how to overcome those problems.
3. Appreciation and identification of suitable learning tasks to which these learning techniques can be applied, and the ability to evaluate how effective a particular learning procedure has been -- internal indicators of learning success vs. actual behaviour of the learner.
4. Use and writing of Matlab programs, ability to set up and run computational experiments to produce statistically sound results
5. Formulation of problems, evaluation of results from the student's own experiments and those presented in some cases in the research literature.

Software Architecture, Process, and Management (Level 11) (INFR11038)

Current Learning Outcomes:

1. Describe and explain the challenges inherent in large-scale system development and outline techniques with which managers can help meet these challenges.
2. Propose and justify architectural decisions for large-scale, long-lived systems.
3. Explain why software reuse is difficult, and some approaches for increasing software reuse.
4. Summarize and apply approaches for maintaining and replacing legacy code.
5. Compare and contrast development processes (e.g. Extreme Programming and the Unified Process) and explain their application to a project.
6. Analyse the significant sources of risk for particular projects and suggest ways to reduce the risks.
7. Describe, and in some cases be able to use, tools relevant to large-scale, long-term development, such as requirements management, configuration, build, test, and project management tools.
8. Critically reflect on given software engineering related articles, from the peer-reviewed literature and elsewhere.
9. Locate, summarize, and critically evaluate peer-reviewed literature about a subarea of software engineering.

Proposed Learning Outcomes:

1. Describe and explain the challenges inherent in large-scale system development and outline techniques with which managers can help meet these challenges, and propose and justify architectural decisions for large-scale, long-lived systems.
2. Explain why software reuse is difficult, and some approaches for increasing software reuse, and summarize and apply approaches for maintaining and replacing legacy code.
3. Compare and contrast development processes (e.g. Extreme Programming and the Unified Process) and explain their application to a project, and analyse the significant sources of risk for particular projects and suggest ways to reduce the risks.
4. Describe, and in some cases be able to use, tools relevant to large-scale, long-term development, such as requirements management, configuration, build, test, and project management tools.
5. Critically reflect on given software engineering related articles, from the peer-reviewed literature and elsewhere, and locate, summarize, and critically evaluate peer-reviewed literature about a subarea of software engineering.

Software Testing (INFR10057)

Current Learning Outcomes:

1. Analyze requirements to determine appropriate testing strategies.
2. Design and implement comprehensive test plans
3. Instrument code appropriately for a chosen test technique
4. Apply a wide variety of testing techniques in an effective and efficient manner
5. Compute test coverage and yield according to a variety of criteria
6. Use statistical techniques to evaluate the defect density and the likelihood of faults.
7. Evaluate the limitations of a given testing process and provide a succinct summary of those limitations
8. Conduct reviews and inspections

Proposed Learning Outcomes:

1. Analyze requirements to determine appropriate testing strategies.
2. Design and implement comprehensive test plans
3. Instrument code appropriately for a chosen test technique
4. Apply a wide variety of testing techniques in an effective and efficient manner
5. Compute test coverage and yield according to a variety of criteria
6. Use statistical techniques to evaluate the defect density and the likelihood of faults.
7. Evaluate the limitations of a given testing process and provide a succinct summary of those limitations
8. Conduct reviews and inspections

Topics in Distributed Databases (INFR11025)

No longer offered

Advanced Databases:

Was: .

1 Demonstrate knowledge of storage methods by enumerating various indexing techniques over single- and multi-dimensional data stored in traditional disk drives, flash memory, and main memory. 2. Demonstrate knowledge of query evaluation by describing and implementing various evaluation algorithms used by database systems in both disk- and non-volatile memory settings. 3. Demonstrate knowledge of cost-based query optimisation by describing search space exploration and different optimisation paradigms. 4. Demonstrate knowledge of transaction processing and concurrency control using lock tables for disk-based, flash memory, and main memory systems. 5. Demonstrate knowledge of crash recovery by describing the methodologies and algorithms employed by a database system in the event of a crash. 6. Demonstrate knowledge of parallel data management by enumerating paradigms of parallel query and transaction processing.

Suggest: combine 3,4

"Demonstrate knowledge of cost-based query optimisation, transaction processing and concurrency control."

Advances in Programming Languages

Was:

1 - Give examples of different programming idioms, other than the imperative class-based object-oriented model which is familiar from Java.
2 - Explain distinctive features of programming idioms, illustrating some relative advantages and disadvantages.
3 - Describe requirements and constraints in the design of programming languages and individual language features.
4 - Outline some of the problems arising from feature interaction in programming languages.
5 - For a range of programming language features, identify the problem they were created to solve, explain the approach they take to do this, and discuss possible problems that may arise.
6 - Describe in depth a specific recent programming language innovation, explaining its motivation, implementation, and how it compares to previous approaches.
7 - Write working code that demonstrates the use of a novel language feature, based on technical research papers and language documentation.

Suggest: merge 1,2; omit 4 (subsumed by 5?),

New 1/2: Give examples of different programming idioms, explain distinctive features and illustrate relative advantages and disadvantages.

Algorithmic Game Theory:

Was:

1 - Understanding of various models of games.
2 - How they are related, and how they arise in various applications in computer science and elsewhere.
3 - An understanding of linear programming and some of its broad applicability.
4 - An understanding of algorithms used to "solve" such games and their efficiency.
5 - Ability to model various scenarios as strategic games, and devise algorithms to solve them.
6 - An understanding of some of the aims of the current research frontier.
7 - Refinement of analytical skills.
8 - The following learning outcomes are all to be demonstrated via a combination of coursework and the exam.

Suggest:

Omit 8, 7 (generic).
Combine 1/2:

> Understanding of various models of games, how they are related, and how they arise in various applications in computer science and elsewhere.

Compiling Techniques: OK

Computational Neuroscience of Vision

Was:

1 - Describe the roles of computational models in biology and informatics
2 - Summarise the basic architecture, development, and known computational functions of early visual areas in humans and monkeys
3 - Search the neuroscientific literature for relevant experimental data
4 - Describe and evaluate different types of computational models
5 - Implement simple models of feature map development and function
6 - Analyse the results of models to make predictions for future experiments

Suggest: combine 3/4

Describe and evaluate different types of computational models
and how they relate to relevant experimental data

Computer Animation & Visualisation

Was:

1 - Describe different representations of 3D objects, and give examples of their application;
2 - Discuss structural properties of data and the influence they have on choice of visualisation algorithm;
3 - Describe a selection of different visualisation algorithms, and explain their features;
4 - Implement a simple visualisation application using an object-oriented visualisation toolkit;
5 - Explain the kinds of animation that can be generated from the application of inverse kinematics or spacetime constraints to control characters;
6 - Describe a range of character animation techniques, and give examples of their application;
7 - Synthesize animations of characters moving around, changing their facial expressions and controlling them in crowds;
8 - Use physically-based simulations to animate the movements of clothes, fluids and particles.

Suggest: merge 1/2, omit 3, 6/7

1/2: Describe different representations of 3D objects, structural properties of data and how these relate to choice of visualisation algorithms.

6/7: Describe and synthesise character animation techniques, including motion, changing their facial expressions and crowd behaviour.

Computer Design

Was:

1 - Build state machines to implement a circuit or system to a specification.
2 - Interconnect circuits for systems of higher complexity, specifically up to the complexity of the components required in a simple computer processor datapath.
3 - Analyse and synthesise circuits to control and sequence the flow of data within a simple cpu or microcontroller.
4 - Analyse and synthesise circuits to control and sequence the flow of data between a simple cpu, memory systems and input/output device controllers.
5 - Design and implement a microprogrammed controller for a given simple cpu architecture.
6 - Gain familiarity with: design and simulation software; designing systems with Verilog HDL; programming designs into a large field-programmable gate array device (FPGA); using an assembly language to implement a design in a programmable microcontroller.

Suggest:

merge 3/4

3/4: Analyse and synthesise circuits to control and sequence the flow of data within a simple CPU or microcontroller, and between a simple CPU, memory systems and input/output device controllers.

Computer Networking

Was:

1 - Be able to describe different types of internet applications (including delay sensitive multimedia applications) and their associated transport and network resource allocations issues.
2 - Be able to contrast between different types of wireless networks in terms of their application scenarios, architectures, resource allocation problems as well as describe cross-cutting issues such as mobility support.
3 - Be able to describe various network performance metrics and their measurement methodologies and techniques.
4 - Be able to grasp the research on advanced topics in networking and summarise it in writing.
5 - Have obtained hands-on experience in simulation based performance study of networks.
6 - Have developed an insight into cutting edge issues in the field of networking.

suggest: merge 4/6

4/6: Have developed an insight into cutting edge issues by understanding the research on advanced topics in networking and and summarising it in writing

Data Integration and Exchange

Was:

1 - Given a description of a situation requiring data integration students will be able to classify it as belonging to one of a number of standard scenarios.
2 - Given a description of data integration and exchange scenario and a commercial database management solution to the problem students will be able to predict erroneous or counterintuitive answers that will arise from the solution. Students will be able to compare & evaluate different solutions & justify the choice of one solution over another.
3 - Given a description of a data integration & exchange scenario where access to data is restricted, students will be able to design a scheme to provide approximate results to queries taking account of restricted access. Students will be able to generate a range of design options & will know techniques and their limitations for comparison between options.
4 - Given a database query & a data integration & exchange scenario, students will know how to rewrite the query to take account of the scenario, providing approximate solutions where these are necessary. Students will be able to develop rewritings of the query & will be able to reason about basic properties of such rewritings.
5 - Given two database schemas students will be able to specify mappings between schemas & how to compose such mappings.
6 - Given a collection of databases, their schemas & mappings between schemas students will be able to describe how data is exchanged based on the schema mappings.
7 - Given a collection of databases, their schemas & schema mappings, students will be able to identify instances of inconsistent data.
8 - Given a collection of databases with inconsistent data, their schemas and schema mappings, students will be able describe the results of queries on inconsistent databases.

Suggest: merge 1/2 and 6/7/8

1/2: Given a description of a situation requiring data integration,
students will be able to classify it as corresponding to a standard scenario,
and to compare and evaluate possible commercial database management solutions.

6/7/8: Given a collection of databases, their schemas & mappings between schemas
students will be able to describe how data is exchanged, to identify instances of
inconsistent data, and describe the results of queries on inconsistent databases.

Discrete Mathematics and Mathematical Reasoning

Was:

1 - Reason mathematically about basic (discrete) structures (such as numbers, sets, graphs, and trees)used in computer science.
2 - Use of mathematical and logical notation to define and formally reason about mathematical concepts such as sets, relations, functions, and integers, and discrete structures like trees, graphs, and partial orders;
3 - Evaluate elementary mathematical arguments and identify fallacious reasoning
4 - Construct inductive hypothesis and carry out simple induction proofs;
5 - Use graph theoretic models and data structures to model and solve some basic problems in Informatics (e.g., network connectivity, etc.)
6 - Prove elementary arithmetic and algebraic properties of the integers, and modular arithmetic, explain some of their basic applications in Informatics, e.g., to cryptography.
7 - Compare the asymptotic growth growth rates of basic functions; derive asymptotic bounds, and limits, for simple series and recurrence relations. Use these to derive bounds on the resource consumption (e.g., running time) of simple iterative and recursive algorithms.
8 - Calculate the number of possible outcomes of elementary combinatorial processes such as permutations and combinations.
9 - Be able to construct discrete probability distributions based on simple combinatorial processes, and to calculate the probabilities and expectations of simple events under such discrete distributions.

Suggest: omit 1 (subsumed by 2?), omit 3 (subsume by 2?),
merge 2/4, 6, omit 8 (implicit in 9?)

2/4:

Use of mathematical and logical notation to define and formally reason about mathematical concepts such as sets, relations, functions, and integers, and discrete structures, including proof by induction.

Honours Project (Informatics)

Was:

1 - Structure and summarise a body of knowledge relating to a substantial project topic in Informatics.
2 - Critically evaluate previous work in the area.
3 - Conduct a programme of work in further investigation of issues related to the topic.
4 - Discuss and solve conceptual problems which arise during the investigation.
5 - Justify design decisions made during the investigation.
6 - Critically evaluate the investigation.
7 - Present their work orally and visually, with demonstration of working artifacts where appropriate.

Suggest:

merge 1/2, and 5/6

1/2: Structure, summarise and critically evaluate a body of knowledge
relating to a substantial project topic in Informatics

5/6: Critically evaluate the investigation, including design choices made.

Informatics 1 - Computation and Logic

Was:

1 - Design a small finite-state system to describe, control or realise some behaviour.
2 - Evaluate the quality of such designs using standard engineering approaches.
3 - Apply the algebra of finite automata to design systems and to solve simple problems on creating acceptors for particular languages.
4 - Describe simple problems using propositional logic.
5 - For a given formula in propositional logic, draw a truth table for that formula and hence deduce whether that formula is true or not.
6 - Apply a system of proof rules to prove simple propositional theorems.
7 - Describe the range of systems to which finite-state systems and propositional logic are applicable and be able to use the meta theory to demonstrate the limitations of these approaches in concrete situations.

Suggest: merge 1/2, merge 4/5

1/2: Design a small finite-state system to describe, control or realise some behaviour, and evaluate the quality of such designs using standard engineering approaches.

4/5: Used propositional logic to describe simple problems, and truth table methods to determine truth values for given propositions.

Informatics 1 - Functional Programming

Was:

1 - Solve simple programming tasks (for example, convert a number into a string for the corresponding roman numeral).
2 - Define appropriate data types (for example, to represent parse trees for arithmetic expressions).
3 - Perform case analysis, use recursion (for example, evaluate a parse tree for an arithmetic expression to yield a value).
4 - Read and write programs that use basic list processing functions (nil, cons, append, length, take, drop, zip, concat).
5 - Read and write programs that use list comprehensions and higher-order functions (map, filter, fold).
6 - Choose appropriate decompositions of problems to create a program to solve that problem.
7 - Compose a functional program from suitable function definitions, including their types.
8 - Document programs effectively.
9 - Apply basic techniques to test and debug programs.

Suggest:

merge 4/5, 6/7 and 8/9

4/5: Read and write programs that use basic list processing functions, list comprehensions and higher-order functions.
6/7: Choose appropriate decompositions of given problems and compose corresponding functional programs from suitable function definitions, including their types.
8/9: Document, test and debug programs.

Informatics 2A - Processing Formal and Natural Languages

Was:

1 - Demonstrate knowledge of the relationships between languages, grammars and automata, including the Chomsky hierarchy;
2 - Demonstrate understanding of regular languages and finite automata;
3 - Demonstrate understanding of context-free languages and pushdown automata, and how how context-free grammars may be used to model natural language;
4 - Demonstrate knowledge of top-down and bottom-up parsing algorithms for context-free languages;
5 - Demonstrate understanding of probabilistic finite state machines and hidden Markov models, including parameter estimation and decoding;
6 - Demonstrate awareness of probabilistic context-free grammars, and associated parsing algorithms;
7 - Demonstrate knowledge of issues relating to human language processing.

Suggest: merge 2/3, 5/6

2/3: Demonstrate understanding of regular languages, finite automata, context-free languages and pushdown automata, and how how context-free grammars may be used to model natural language.

5/6: Demonstrate understanding of probabilistic finite state machines and hidden Markov models, including parameter estimation and decoding, and probabilistic context-free grammars, with associated parsing algorithms.

Informatics 2C - Introduction to Computer Systems

Was:

1 - Demonstrate an understanding of binary representation and basic operations on binary data.
2 - Demonstrate an understanding of key concepts in computer architecture, including: exceptions, interrupts, virtual memory, processes and pipelined execution.
3 - Sketch the design of a simple processor and explain how it operates.
4 - Demonstrate knowledge of I/O devices and the means by which they interface to a processor and its memory system.
5 - Demonstrate an understanding of the design and operation of important combinational and sequential components within a processor, such as adders, registers, and state machines.
6 - Demonstrate understanding of an execution pipeline, based on the MIPS architecture.

Suggest: merge 1/2

1/2 Demonstrate an understanding of key concepts in computer architecture, including binary representation, exceptions, interrupts, virtual memory, processes and pipelined execution.

Informatics 2D - Reasoning and Agents

Was:

1 - Use task constraints to make search efficient
2 - Perform Inference with First Order Logic
3 - Comprehend the strengths and weaknesses of various kinds of
logic representations, e.g. Propositional, FOL
4 - Use STRIPS to plan and execute actions using either Propositional or
First Order Logic representation.
5 - Create a Bayesian net representation of a non-deterministic planning problem
6 - Create a basic probabilistic action agent using simulated state transitions and goals

Suggest: merge 2/3

2/3: Perform Inference with First Order Logic and appreciate
the strengths and weaknesses of this and other
logic representations (eg Propositional)

Introduction to Java Programming

Was:

1 - Students will be able to state, in writing and verbally, basic principles of object-oriented software design.
2 - Given an object oriented design as a diagram or textual description, students will be able to evaluate the quality of that design and discuss its strengths and weaknesses with respect to its stated purpose.
3 - Students will be able to design an object oriented software solution to a problem using diagrammatic and textual representations.
4 - Students will be able to implement an object oriented design in the Java language.
5 - Students will be able to relate the syntax of the Java language to its semantics, and analyse the result of executing fragments of Java syntax.
6 - Given a Java program, students will be able to explain, in writing and verbally, what would happen when that program is executed, and identify bugs which would prevent it executing as described in the program documentation.
7 - Given a Java program and a debugging tool, students will be able to identify and correct bugs which prevent the program from functioning as intended.
8 - Students will be to write documentation in Javadoc style to explain the design and implementation of their own code, or example code which is supplied to them.
9 - Students will be able to use an appropriate software development environment, such as BlueJ, Eclipse or NetBeans.
10 - Students will be able to integrate library code with their own programs using appropriate software tools.
11 - Students will be able to use online technical documentation to solve implementation problems as they arise during software development.
12 - Students will be able to describe stages in the software development process and the identify software tools which are used to support these stages.

Suggest:

1 - Students will understand and be able to articulate basic object-oriented design principles, use them to design solutions to given problems, evaluate the quality of such designs, and implement designs in Java.

2 - Students will be able to explain the connection between Java syntax and semantics, to describe behaviour expected from given code, and to identify and correct bugs.

3 - Students will be to write documentation in Javadoc style, make use of appropriate software development environments, integrate library code with their own programs and make appropriate use of online technical documentation.

4- Students will be able to describe stages in the software development process and the identify software tools which are used to support these stages.

MInf Project (Part 1)

As for Honours Project

MSc Dissertation

As for Honours Project

MSc by Research Thesis (Data Science; 120pt)

As for Honours Project

MLPR

Recently revised, OK

Natural Language Understanding (Level 11)

Was:

1 - Given a parsing problem students should be able to use state-of-the-art symbolic parsing techniques, including lexicalised parsing to solve the problem and provide a written explanation of the parsing techniques used in the course.
2 - Given a labelled corpus, students should be able to select and use state-of-the-art statistical parsing techniques (generative and discriminative) by training parsers on the labelled corpus using existing software packages.
3 - Given an NLU system, students should be able to choose appropriate evaluation metrics for the system, and use error analysis to propose improvements to the language processing models.
4 - Given an example of a problem in coreference resolution, discourse segmentation, and discourse parsing, students should be able to provide a written description of how current symbolic and statistical techniques help solve the problem.
5 - Given a description of an NLU system, the student should be able to relate it to features of human models of language interpretation at various levels of processing (words, sentences, discourse and dialogue).
6 - Given a model and a labelled corpus, students should be able to employ existing ML software packages to train the model on the corpus in order to perform a lexical semantic task.
7 - Given an open-ended problem of choosing informative features for a particular NLP task and a description of the available training resources, the student should be able to give a well-justified, written and/or practical, selection of such informative features.

Suggest:

merge 1/2, 3/5

1/2: Students should be able to use and explain appropriate
state-of-the-art symbolic parsing techniques, and, where a labelled corpus
is available, statistical parsing techniques (generative and discriminative).

3/5: Given an NLU system, students should be able to choose appropriate
evaluation metrics for the system, use error analysis to propose improvements,
and relate it to features of human models of language interpretation at
various levels of processing.

Pervasive Parallelism

Was:

1. Select literature appropriate for the review subject.
2. Critically evaluate research literature in the chosen area.
3. Search and use appropriately databases of scientific literature.
4. Evaluate and search traditional library resources.
5. Discuss a research topic in detail leading to new hypotheses.
6. Deliver a brief but balanced report on a research topic.
7. Critically evaluate research literature appropriate for their project subject.
8. Use existing research literature to justify experimental design choices.
9. Develop a structured research proposal.
10. Discuss research proposals with particular reference to key hypotheses and methodological approaches.
11. Awareness of project/research management issues.

Suggest:

1. Make use of a range of resources (on-line, library) to gather, analyse and critically review literature on the review subject.
2. Discuss a research topic in detail, come up with new hypotheses and deliver a corresponding report.
3. Use existing research literature to justify experimental design choices.
4. Develop a structured research proposal.
5. Analyse research proposals with particular reference to key hypotheses and methodological approaches, and show awareness of projectt/research management issues.

Professional Issues (Level 10)

Was:

1 - describe the desirable attributes of graduates in computing
2 - explain the importance of professionalism in computing
3 - identify how and where social and ethical implications arise in computing
4 - describe the legal issues that impact computing
5 - describe the structure and operation of commercial computing organisations
6 - describe and explain the relationships between scientific,
technical & engineering issues and real world issues in computing
7 - construct a well-written essay

Suggest: merge 1/2, 3/4

1/2: describe the desirable attributes of graduates in computing,
and explain the importance of professionalism in computing.

2/4: identify how and where social, ethical and legal
issues relate to computing.

Robotics and Autonomous Systems Research Thesis

Was:

1.Gain basic skills in a robotics area of choice.
2.Structure and summarise a body of knowledge relating to a substantial project topic in Robotics and Autonomous Systems.
3.Critically evaluate previous work in the area.
4.Conduct a programme of work in further investigation of issues related to the topic.
5.Discuss and solve conceptual problems which arise during the investigation.
6.Justify design decisions made during the investigation.
7.Critically evaluate the investigation.
8.Present their work, with demonstration of working artefacts where appropriate.

Suggest:

merge 2/3, 4/1, 6/7

2/3: Structure, summarise and critically evaluate a body of knowledge relating to a substantial project topic in Robotics and Autonomous Systems.

4/1: Gain basic skills in a robotics area of choice, and conduct a programme of work in further investigation of the project topic.

6/7: Critically evaluate the investigation, including design choices made.

SELP, part of LP package

System Design Project

Was:

1 - Working as members of a team.
2 - Planning and monitoring the effort of a project to meet milestones and deadlines.
3 - Designing and implementing a complex and multi-faceted system.
4 - Attempting to achieve a difficult objective within a limited time-scale.
5 - Drawing together knowledge and understanding of wide areas of
software and hardware systems.
6 - Demonstrating and presenting the outcome from a practical project.
7 - Documenting the feasibility, design and development of a potential product.

Suggest:

merge 1/3, 2/4

1/3: Working as members of a team in designing and implementing a
complex and multi-faceted system.

2/4: Planning and monitoring the effort of a project to meet milestones and deadlines,
within a limited time scale.